Centro Universitário de Brasília

DANIEL RIBEIRO DE MOURA

OCULTAÇÃO DE TEXTO CRIPTOGRAFADO EM IMAGEM JPEG

Brasília/DF, Novembro de 2006.

DANIEL RIBEIRO DE MOURA

OCULTAÇÃO DE TEXTO CRIPTOGRAFADO EM IMAGEM JPEG

Monografia apresentada ao Centro Universitário de Brasília – UniCEUB como um dos pré-requisitos para obtenção do título de Bacharel em Engenharia da Computação.

Profº Orientador: MC Aderlon M. Queiroz

Brasília/DF, Novembro de 2006.

Agradecimentos

A Deus,

Por ter me dado força de vontade.

Aos meus pais,

Por sempre me incentivarem.

Aos meus familiares,
Por me apoiarem.

Ao Professor Francisco Javier,
Pela ajuda, paciência, colaboração e apoio durante todos esses anos, não só no projeto final, mas sim desde quando o conheci no 6° semestre.

Ao meu orientador Mcs. Aderlon M. Queiroz,
Que me ajudou quando precisei.

A todos,
O meu gesto de gratidão.

RESUMO

Este trabalho apresenta a implementação de um aplicativo com uma interface gráfica, capaz de ocultar/extrair um texto criptografado em uma imagem no formato JPEG, utilizando-se da técnica de esteganografia, em conjunto com o recurso criptográfico MD5 (Message Digest 5), desenvolvido no ambiente e linguagem MatLab, para ocultação do texto na imagem, será utilizada a técnica do LSB (Least Significant Bit) bit menos significativo, de cada pixel da imagem. Por fim o aplicativo apresenta uma verificação da imagem esteganografada, através de histogramas e recursos estatísticos a fim de averiguar se a imagem sofreu perda de qualidade, após a inclusão do texto.

Palavras chaves: esteganografia, MD5 (Message Digest 5), bit menos significativo, formato JPEG.

ABSTRACT

This work presents the implementation of a program with a graphical interface which is capable of hiding or extracting a cryptographic text in a JPEG image by using the technique of steganography. Together with this technique, there is a cryptographic resource called MD5 (Message Digest 5) that was developed in the MATLAB environment and language. Moreover, there is another technique which is going to be used for hiding the text into the image called Least Significant Bit. The Least Significant Bit will be working in each pixel of the image. Finally, the program will show the stego image through statistics resources and histograms and also it will check if the image has lost quality after the text insertion.

Keywords: *steganography, MD5 (Message Digest 5), least significant bit, JPEG format.*

Sumário

Lista de Figuras

Lista de Abreviaturas e Siglas

BIT	*Binary Digital* (Menor unidade de informação binária)
GIF	Graphics Interchange format(Formato de Imagem)
JPEG	*Joint Photographic Experts Group* (Formato de Imagem)
LSB	*Least Significant Bit* (Bit menos significativo)
LZW	*Lempzel-Ziv-Welch* (Algoritmo de compressão de imagem)
PIXEL	*Picture Element* (Elemento da imagem)
RGB	*Red, Green, Blue* (Vermelho, Verde, Azul)
TIFF	*Tagged Image File Format* (Formato de Imagem)

1. INTRODUÇÃO

1.1. Motivação

Com o expressivo crescimento da troca de informação, e dos possíveis meios em que podem ser transmitidas, principalmente com o surgimento e avanço tecnológico da Internet, a segurança torna-se indispensável para a transmissão da informação, pois existem várias maneiras de interceptar essas informações trocadas. E pensando na segurança da informação que é transmitida, criaram mecanismos de tornar a troca de informação mais segura, como é o caso da esteganografia, que entre diversos benefícios, possibilita a troca de informações sigilosas de forma segura e eficiente, dessa forma, favorecendo o aumento da privacidade e da confidencialidade de informações particulares de trafegam em um meio comum, como por exemplo, uma rede local ou na Internet.

Outra maneira de deixar a troca da informação mais segura é a criptografia, que se aliada ao processo de esteganografia, aumenta o nível de segurança, pois, se uma mensagem for criptografada e, em seguida, esteganografada, será extremamente difícil quebrar esse código, pois primeiro será necessário descobrir a existência da mensagem, para então depois tentar quebrá-lo.

1.2. Objetivo

O trabalho proposto tem como objetivo desenvolver um aplicativo com uma interface gráfica, capaz de ocultar/extrair um texto criptografado em uma imagem estática 2D no formato JPEG, utilizando a técnica de esteganografia aliada ao recurso criptográfico MD5. Implementando também, uma verificação da imagem esteganografada, averiguando se ocorreu perda de qualidade, isto é, se a imagem não foi degradada, após a inclusão do texto, através de histogramas e dados estatísticos para se ter uma comparação visual e mais precisa.

1.3. Metodologia

Para o desenvolvimento do trabalho a ferramenta a ser utilizada para implementação do projeto, será o MatLab 6.5 (Release 13), por ser uma excelente plataforma de desenvolvimento, que possui diversas ferramentas implementadas para as mais diversas aplicações, e também por apresentar um ambiente rico para visualização de dados devido a sua poderosa capacidade gráfica.

Como recurso criptográfico será utilizado o MD5 (Message Digest 5), que produzirá um hash ou resumo digital de 128 bits, a partir de uma senha qualquer (string), e através da função lógica ou-exclusivo XOR, esse hash será utilizado para criptografar o texto. Dessa forma, só será possível extrair o texto, se for conhecida a senha, isto é, a chave criptográfica. O MD5 foi escolhido por ser uma função unidirecional

extremamente segura, além de ser de fácil acesso, por existir muito material de pesquisa em livros e na internet, inclusive mostrando toda a sua estrutura e o seu funcionamento.

Na ocultação do texto na imagem, será utilizada a técnica do LSB (Least Significant Bit) bit menos significativo, de cada pixel da imagem, por ser um dos métodos mais comuns e utilizados atualmente para esconder informações dentro de imagens, também por possuir um acervo para pesquisa bastante amplo, e fácil de ser encontrado em livros e principalmente na internet.

1.4. Estrutura do trabalho

Após este capítulo introdutório, o Capítulo 2 expõe alguns conceitos e características da imagem digital, conceituando também o formato JPEG. O Capítulo 3 apresenta definições e características de esteganografia, descrevendo as técnicas e métodos de esteganografia utilizados no projeto. No Capítulo 4 é apresentado conceitos e definições de criptografia, também apresenta alguns métodos de criptografia, inclusive o recurso criptográfico utilizado no projeto. O Capítulo 5 apresenta o protótipo desenvolvido. No Capítulo 5 são apresentados os testes, simulações e resultados obtidos. E por fim, expõe as considerações finais e conclusões do trabalho, e também as recomendações para trabalhos futuros.

2. IMAGEM DIGITAL

2.1. Definição

Uma imagem digital $\hat{I}$ pode ser definida como um par (***Di, I***), onde ***Di*** é um conjunto de pontos do Z^n(domínio da imagem), denominados pixels, e ***I***, é um mapeamento vetorial que associa a cada spel ou pixel ***p*** em ***Di*** um conjunto {***Ia***(***p***), ***Ib***(***p***), ***....***, ***Ik***(***p***)} de valores escalares, associados com alguma propriedade física, o valor ***n*** refere-se à dimensão da imagem e o valor de ***k*** ao número de bandas.

Alexandre falcão (2003), define uma imagem como "uma função de intensidade de luz bidimensional *f*(*x*, *y*), onde *x* e *y* denotam coordenadas espaciais e o valor de *f* no ponto (*x*, *y*) é proporcional ao brilho da imagem neste ponto".

2.2. Atributos de uma Imagem

Como já vimos anteriormente, uma imagem é uma matriz de pontos ou pixels, com uma determinada resolução horizontal (eixo X) e vertical (eixo Y), onde para cada ponto desta matriz temos uma cor associada.

2.2.1. Amostragem e Quantização da Imagem

> "Para gerar uma imagem digital, *f(x, y)* deve ser digitalizada ao longo de *x* e *y*, e na amplitude *z* = *f(x, y)*. Para isso é feita uma amostragem (uniforme) de *f(x, y)* nas direções x e y, gerando uma matriz de *N* x *M* amostras, seguida de uma

> quantização do valor de *f(x,y)*. Nesta matriz, cada elemento p(x, y), *x* = 0, 1,...., *M*-1 e *y* = 0, 1,...., *N*-1 é chamado *pixel* (abreviação de picture elements)". (Alexandre Falcão 2003).

Pode-se dizer então que uma imagem tem dimensão M pixels na horizontal e N pixels na vertical, e ainda que o a dimensão de um pixel ao longo dos eixos x e y, está relacionada com o espaçamento físico entre as amostras.

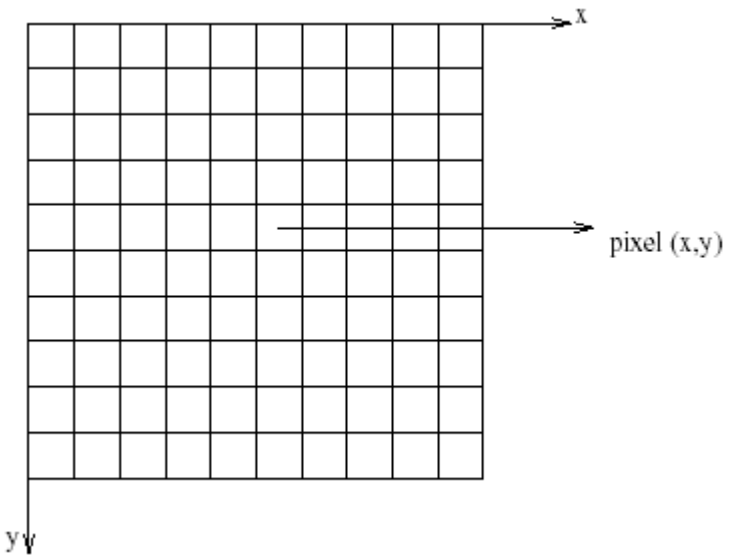

Figura 2.1 – Exemplo digitalização de uma imagem.

Fonte: [FALCÃO 2003].

2.2.2. Cor

A cor é extremamente necessário ao analisar uma imagem, principalmente pela relação com percepção humana, a maioria das cores visíveis pelo olho humano pode ser representada pela combinação de luzes monocromáticas em azul, vermelho e verde. Segundo Alexandre Falcão (2003),

> "O olho humano percebe cerca de 30 níveis de cinza e 7 milhões de cores, ele é mais sensível ao verde, depois ao vermelho, e menos ao azul, porém percebe mais variações de azul, depois de vermelho e menos de verde".

Numa imagem colorida pode ser composta por aproximadamente três grandezas: Luminancia, que está relacionada ao brilho da luz. Matriz, que é responsável pela cor e Saturação, que corresponde pela pureza da cor.

Existem alguns sistemas que descrevem as características da cor, e entre os principais destaca-se o sistema RGB que é muito usado para mostrar imagens coloridas no computador, o espaço RGB é formado pela soma ponderada do red (R), green (G) e blue (B).

Figura 2.2 – Exemplo de imagem RGB, onde (a) RGB - 24 bits, (b) componente R, (c) componente G, (d) componente B.

Fonte: [USC - University of Southern California. *The USC-SIPI Image Database*].

2.3. Imagem JPEG

JPEG é um tipo de arquivo que serve para o armazenamento de imagens, que trabalha com o esquema de 24 bits de cores, aceitando aproximadamente 16,8 milhões de cores, e possui um algoritmo de compactação que se baseia na capacidade do olho humano. Mesmo que o formato de imagem JPEG permita trabalhar com um número extremamente grande de cores, o olho humano não é capaz de enxergar todas de uma vez, então, o formato JPEG retira uma série de informações que representam cores em imagens e mantém apenas aquelas visíveis ao olho humano. "Em outras palavras, o formato JPEG tira da imagem aquilo que os humanos não conseguem ver" (Alecrim 2004). O JPEG (Joint Photographers Expert Group)[1] é um padrão internacional, proposto pelo comitê ISO[2] (International Standard Organization) que permite a transparência de arquivos por uma grande variedade de plataformas, possuindo uma codificação da imagem por transformação matemática, com isso oferecendo altas taxas de compressão, podendo comprimir imagens em até um quinto (1/5) do tamanho original, sem perda de qualidade perceptiva.

O JPEG foi praticamente desenvolvido para que as imagens criadas com milhões de cores pudessem ser exibidas e também para compactar os arquivos de imagem com uma menor ocupação de espaço em disco. Fazendo uma pequena comparação do JPEG com outros formatos,

[1] Nome do grupo que descreveu o padrão JPEG.
[2] Organização Padrão Internacional.

podemos perceber que o método de compressão do JPEG oferece taxas de compressão mais altas do que o LZW usado pelo GIF e pelo TIFF, usando um mecanismo que permite ao usuário alterar um nível de qualidade, lembrando que quanto menor a taxa de compressão, maior é a qualidade da imagem. O JPEG utiliza internamente um procedimento matemático complexo, que examina e agrupa blocos de 8x8 à 64x64 píxels, fazendo uma operação de média dos valores de cor.

2.3.1. Vantagens da Imagem JPEG

Entre as principais vantagens que o formato de imagem JPEG possui, destacam-se:

- Estrutura utilizada por diversas plataformas;
- É um dos formatos de imagem mais popular e utilizado atualmente, principalmente na Internet;
- É superior em comprimir imagens em cores reais ou tons de cinza que retratem cenas reais;
- Trabalha bem em fotografias, artes naturalística e material similar;

2.3.2. Desvantagens da Imagem JPEG

As principais desvantagens do formato de imagem JPEG são:

- Perda de qualidade a cada vez que o arquivo é salvo.

- Não é adequada para imagens simples como faixas, cartões ou desenho com linhas;
- Sofre perda no processo de compressão;

3. ESTEGANOGRAFIA

Jascone (2003), define esteganografia como

> "a arte de comunicar-se secretamente, ocultando uma mensagem sigilosa dentro de outra informação sem importância, de maneira que não exista forma de detectar que há uma mensagem escondida. Na computação essa outra informação pode ser um arquivo de som, imagem ou texto".

A esteganografia consiste em ocultar ou esconder uma determinada informação, seja um arquivo de texto ou uma imagem, dentro de outro arquivo, de forma que sua existência não seja percebida de forma normal. Por exemplo, esconde-se um texto dentro de uma fotografia qualquer e envia a mesma para alguém, ao receber essa foto, não será possível enxergar o texto, somente a fotografia, o texto só descoberto por alguém que faça uso da mesma técnica que foi utilizada para esconder o texto. Estegano, do grego significa "oculto" ou "misterioso" e Grafia, "escrita" ou "desenho".

É importante não confundir esteganografia com criptografia, pois o primeiro esconde a existência da mensagem, já o segundo esconde o conteúdo da mensagem, mas sua existência é conhecida. As duas técnicas podem ser utilizadas em conjunto para se obter um maior grau de segurança da informação. Embora os dois métodos (esteganografia e criptografia) podem ser combinados para aumentar a segurança dos sistemas de informação. Por exemplo, pode-se utilizar a técnica de esteganografia em uma mensagem e em seguida criptografar à mesma, trocando-se os bits menos significativos de uma imagem digitalizada pelos

bits da mensagem criptografada, e então transmitir a imagem, se a imagem for interceptada, primeiro será necessário descobrir a mensagem oculta entre os bits da imagem, e, somente após isso, poderá ocorrer a tentativa de descriptografá-la.

Esta arte de esconder mensagens não é exclusividade da atualidade, há relatos que através da história foram criados vários mecanismos para esconder informações embutidas em outros objetos, visando alcançar privacidade nos meios de comunicação, onde conheceremos alguns a seguir.

Segundo Petticolas *et al* (1999), os primeiros casos que relatam o uso de esteganografia, foi o de uma lebre morta, na qual seria escondida uma mensagem em suas entranhas, e que seria enviada até o seu destino. Outro método utilizado era o do raspar a cabeça dos mensageiros e desenhar a mensagem que seria transmitida, e depois de crescido o cabelo do mensageiro, o enviava até o destino.

Já Rocha (2003), diz que uma técnica bastante utilizada era o uso de tabletes de madeira cobertos de cera.

> "Os tabletes eram utilizados como meio de escrita para a época, na Grécia antiga. Os textos eram escritos sobre a cera e, quando não eram mais utilizados, a cera era derretida, e era colocada sobre a madeira uma nova camada de cera, gerando outro tablete de cera novo e pronto para a escrita. O método consistia em escrever a mensagem diretamente na madeira e depois cobri-las com cera, parecendo dessa forma um tablete de cera normal, com o detalhe de estar ocultando uma mensagem".

Somente no século XV é que o termo "esteganografia" passou a ser conhecido, após a publicação de uma trilogia intitulada de

Steganographia, do monge Trithemius, que detalhava várias técnicas para passar mensagens de forma desapercebida.

De acordo com Jascone (2003),

> "Na Segunda Guerra Mundial, os métodos esteganográficos se baseavam na utilização de micropontos, devido ao aumento da qualidade das câmeras, lentes e filmes. Por meio deste método uma mensagem secreta poderia ser fotografada e reduzida ao tamanho de um ponto, e podendo este ser um ponto final (.) de sentença ou ponto de uma letra (i) de uma outra mensagem qualquer".

Atualmente os hackers e crackers também passaram a inovar nas formas de trocarem informações, e uma das técnicas mais utilizadas por eles é sem dúvidas a esteganografia digital, utilizada para esconder informações em textos, imagens, sons e vídeos, de forma que as mesmas passem desapercebidas aos olhos humanos.

3.1. Esteganografia Digital

A esteganografia atual oculta informações dentro de arquivos de áudio, vídeo ou imagens, onde a informação é escondida nos bits menos significativos dos bytes do arquivo. A mensagem esteganografada só poderá ser lida por quem souber onde ela está e se conhecer o código para decifra-la.

As técnicas de esteganografia digital também buscam a ocultação da informação ou, ao reverso, como neutralizar essa possibilidade. Muito do desenvolvimento da esteganografia digital aconteceu em função da necessidade de proteção da propriedade intelectual.

> "A forma mais comum e utilizada para esconder informações ainda é através de imagens. Arquivos digitais, de maneira geral, possuem áreas não utilizadas, ocupáveis por informação adicional. Uma imagem é formada por um conjunto de pixels de 8 bits. O pixel é a unidade básica de programação de cor em um display de computador ou em um arquivo de imagem. A cor específica que um pixel descreve é uma mistura dos três componentes do espectro de cores – vermelho, verde e azul". (Pinheiro 2005).

3.2. Aplicações

A esteganografia possui diversas aplicações, mas todas com objetivo de alcançar a privacidade das suas informações. Em Gois (2003), encontramos algumas aplicações na indústria, para esconder informações, ou seja, para a proteção da privacidade, conforme será visto a seguir.

3.2.1. Proteção de Direitos Autorais

Quando um autor de uma imagem, vídeo, som ou texto deseja assinar sua obra para que nenhuma pessoa possa atribuir-se ou aproveitar-se das obras de sua autoria.

3.2.2. Certificação e Controle de Acesso

Utilizado em documentos oficiais, por exemplo, carteiras de identidade e passaportes, permite validar as informações contidas em um documento através da sua vinculação.

3.2.3. Legenda em Imagens e Vídeos

Neste caso as legendas seriam escondidas em imagens ou vídeos de modo que as mesmas possam ser extraídas e apresentadas. Este método também pode ser aplicado em som ou informações adicionais de um filme de DVD, por exemplo.

3.3. Técnicas de Esteganografia

Existem diversos métodos para aplicar a esteganografia, as técnicas mais comuns e utilizadas atualmente são:

- Esteganografia em Áudio;
- Esteganografia sobre TCP/IP;
- Esteganografia em Imagens;

Neste capítulo mostramos mais detalhado como funciona o método de esteganografia em imagens.

3.3.1. Esteganografia em Áudio

Para ocultar dados em áudio, é necessário explorar as fraquezas do sistema auditivo humano (SAH). O Sistema Auditivo Humano (SAH) percebe uma faixa de freqüência muito grande, sendo também muito sensível ao ruído. Qualquer tipo de alteração em um arquivo de áudio pode ser detectado. Os principais métodos utilizados para esconder dados em áudio segundo Bender (1996), são:

3.3.1.1. Inserção no *bit* menos significativo

Dados binários podem ser armazenados nos *bits* menos significativos dos arquivos de áudio. A capacidade do sinal é de 1 *Kb* por segundo por *kilohertz*, então, por exemplo, em uma seqüência de reprodução de 44 *kHz* a capacidade do sinal seria de 44 *kbps*. Infelizmente, isto introduz ruído audível. Uma das desvantagens deste método é a pequena imunidade a manipulações. Fatores como ruído de canal e regravação podem facilmente destruir o sinal oculto.

3.3.1.2. Esconder dados no eco do áudio

Nesta técnica são inseridos os dados em sinal próprio gerando um eco. Os dados são ocultados através da variação de três parâmetros de eco: amplitude inicial, taxa de deterioração e o atraso. Em algum ponto, a audição humana não pode distinguir entre o som original e o eco, onde o sinal do eco é ouvido meramente como ressonância.

3.3.2. Esteganografia em TCP/IP

Este método oculta informações dentro do cabeçalho de um pacote TCP/IP. A arquitetura básica do pacote TCP/IP permite um número de opções secretas, ou seja, a utilização dos campos não usuais ou opcionais que ficam dentro dos pacotes. Exemplos deste método podem ser vistos em Owens (2002).

3.3.3. Esteganografia em Imagens

Essencialmente, aplicar a esteganografia em imagens é saber explorar os poderes limitados do sistema visual humano. A esteganografia em imagem evoluiu muito com o desenvolvimento de computadores gráficos rápidos e poderosos, além de que os softwares esteganográficos estão disponíveis para usuários diários da Internet, que, aliás, é um canal vasto para disseminação de informação, imagens são utilizadas em toda a rede com diversos propósitos, elas provêem excelentes recipientes para esconder informações.

Existem várias formas para esconder informações em imagens, as mais comuns são:

- Inserção no bit menos significativo;
- Máscara e filtragem;
- Algoritmos e transformações.

O projeto proposto utiliza o método do LSB (inserção no bit menos significativo), a idéia é que, alterando-se o bit menos significativo não ocorrem mudanças perceptíveis na imagem, dessa forma possibilitando codificar em uma imagem uma seqüência de dígitos binários que contenha um texto usando apenas o bit menos significativo de cada componente da cor dos pixels, como mostra a seguir.

3.3.3.1. Inserção no *bit* menos significativo

> "O método Least Significant Bit (LSB) é o mais comum utilizado para armazenar informação em imagens digitais. Consiste em utilizar o *bit* menos significativo de cada *pixel* (ou de cada cor) da imagem, para ocultar a mensagem". (Jascone 2003).

Jascone (2003), também mostra um exemplo demonstrando a utilização do método do *bit* menos significativo, que assim explica:

> "para esconder uma informação em uma imagem de 24 *bits*, onde cada *pixel* possui 3 *bytes*, utilizando o método de inserção do último *bit* significativo, pode-se armazenar 3 *bits* em cada *pixel*, utilizando 1 *bit* de cada *byte* desse *pixel*, ou seja, cada *byte* da informação a ser escondida irá ocupar 8 *bytes* da imagem. Por exemplo, em uma imagem de 24 *bits* com resolução de 1024 x 768 (768.432 *pixels*), que possui um tamanho real de 2.359.296 *bytes* (cada *pixel* ocupa 3 *bytes*), pode-se armazenar uma informação de 294.912 bytes (8 *bytes* da imagem para armazenar 1 *byte* do texto)".

A figura 3.1, abaixo mostra que para armazenar a letra "A", que possui um valor binário de 0 1 0 0 0 0 0 1. Inserindo esse valor binário, *bit* a *bit*, na imagem, utilizando o *bit* menos significativo de cada *byte* do *pixe*l, onde o resultado é apresentado na figura 3.2 . Os valores (*bits*) sublinhados são os que sofreram alterações, causando uma imperceptível alteração na cor do *pixel*, para o olho humano é impossível distinguir esta diferença. Esses valores combinados formam o *byte* que representa a letra "A".

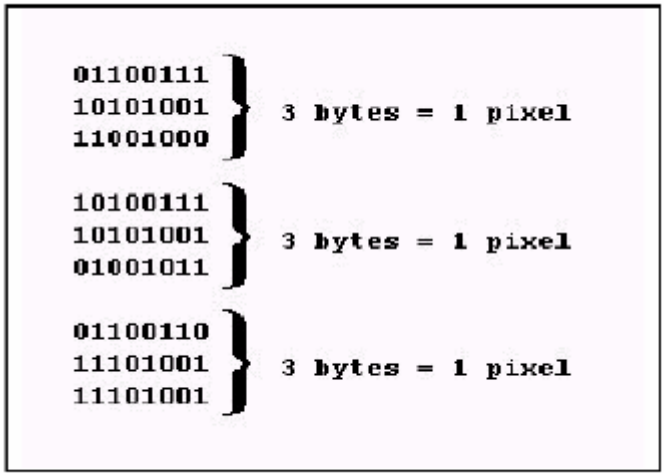

Figura 3.1 – Exemplo de pixels de uma mensagem

Fonte: [JASCONE 2003]

A imagem resultante seria como mostra a figura abaixo.

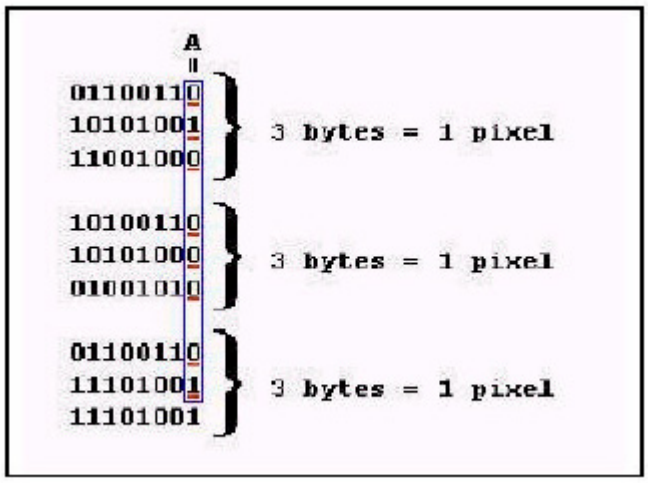

Figura 3.2 – Exemplo de uso do método LSB.

Fonte: [JASCONE 2003]

3.3.3.2. Máscara e filtragem

"Técnicas de filtragem e mascaramento são restritas a imagens em tons de cinza [...]. Estas técnicas escondem a informação através da marcação de uma imagem, semelhante ás marcações de *copyright* em papel". (Rocha 2003).

Jascone (2003), afirma que a marca d'água é integrada na imagem, desta forma podemos destacar como uma vantagem, a possibilidade de aplicar métodos de compressão nas imagens. Outra vantagem é que essas técnicas passam despercebidas pelo sistema visual humano que não conseguem distinguir algumas mudanças na imagem.

3.3.3.3. Algoritmos e transformações

De acordo com Rocha (2003), os algoritmos de transformação trabalham com formas mais sofisticadas de manuseio de imagens, entre elas estão: o brilho, a saturação e a compressão das imagens. As técnicas de transformação tomam como aliado o principal inimigo da inserção *LSB*, a compressão. Por isso tornam-se como as mais sofisticadas técnicas de mascaramento de informações em imagens conhecidas.

4. CRIPTOGRAFIA

A criptografia pode ser definida como a arte de manter mensagens secretas. "Criptografia vem da palavra grega kryptos = escondida e graphia = escrever" (Oliveira, 2001). A criptografia consiste basicamente em um processo de cifragem ou codificação para converter o texto claro em texto cifrado, ou em um processo inverso que é a decifragem, onde se obtém o texto claro, ou seja, o texto original a partir do texto cifrado. Os processos de cifragem e decifragem de uma mensagem são realizados de acordo com algoritmos criptográficos, estes algoritmos são funções matemáticas usadas para cifrar e decifrar. Estes algoritmos dependem de um pedaço adicional de informação, chamado de chave, onde esta chave é combinada com o texto em claro para obter o texto cifrado ou vice-versa.

Criptografar quer dizer transformar uma mensagem em outra, escondendo a mensagem original, com a elaboração de um algoritmo com funções matemáticas e uma senha especial, chamada chave. Para que um algoritmo criptográfico seja considerado eficiente e seguro, é necessário publica-lo, para que seja realizada uma criptnálise[3] sobre o algoritmo, por isso, diz-se que a segurança não está nos detalhes dos algoritmos e sim na manutenção do segredo da chave utilizada.

Oliveira (2001), coloca que

> "a chave consiste em uma *string* que pode ser alterada sempre que necessário. Desse modo, o algoritmo de

[3] Tentativa de descobrir chaves criptográficas.

> criptografia pode ser conhecido. Quando o algoritmo se torna público, vários especialistas tentam decodificar o sistema. Se, após alguns anos, nenhum deles conseguirem a proeza, significa que o algoritmo é bom".

Para decodificar uma chave é necessária uma pesquisa no espaço da chave, que é exponencial em relação ao tamanho da chave, com isso quanto maior for a chave, maior será o trabalho que o intruso terá para decodificar a chave. Por exemplo, uma chave de dois dígitos existem cem possibilidades, e com um tamanho de chave de seis dígitos, existem um milhão de possibilidades.

Na criptografia encontram-se serviços que asseguram a confidencialidade, autenticação e integridade dos dados, conforme definido abaixo:

- Confidencialidade

Restringe o acesso aos dados somente ás pessoas autorizadas, ou seja, manter o conteúdo da informação protegida.

- Autenticação

Estabelece a autenticidade e validade da transmissão, da mensagem e das entidades envolvidas.

- Integridade

Garante que os dados recebidos sejam idênticos aos dados enviados e que não sofram nenhuma alteração acidental ou não autorizada.

Na história da criptografia há relatos de que ela já era utilizada na época da escrita hieroglífica. Algumas nações investiram muito na criptografia durante a primeira e segunda guerra mundial para transmitir

informações secretas e tirar alguma vantagem dos seus adversários com o uso da criptografia.

A máquina criptogáfica alemã "Enigma" utilizada na segunda guerra mundial, foi um dos marcos mais importantes da história da criptografia. O "Enigma" parecia com uma máquina de escrever, e possui rotores e dependendo da posição que estes eram colocados utilizavam um determinado algoritmo.

A criptografia começou a crescer com as pesquisas científicas realizadas pelas universidades e agências do governo americano. A partir desse período, com advento dos computadores, a criptografia se deslanchou e passou por várias mudanças nos métodos empregados, onde milhares de cálculos podiam ser realizados em questão de segundos. Esse novo método utiliza algoritmos matemáticos mais complexos, com os quais foram aumentando cada vez mais o grau de dificuldade para serem descobertos, como é o caso dos algoritmos utilizados atualmente.

Atualmente os métodos de criptografia conhecidos e utilizados são, criptografia de chave privada, criptografia de chave pública e função de hash.

4.1. Criptografia de Chave Privada

A criptografia de chave privada também é conhecida como criptografia simétrica, este método utiliza o uso da mesma chave tanto para encriptar como para decriptar uma mensagem trocada. Porém para

assegurar a eficiência deste método é necessário que a chave seja conhecida somente pelo emissor e receptor e utilizem o mesmo algoritmo.

A figura 4.1 ilustra um modelo de funcionamento de criptografia baseada em chave simétrica.

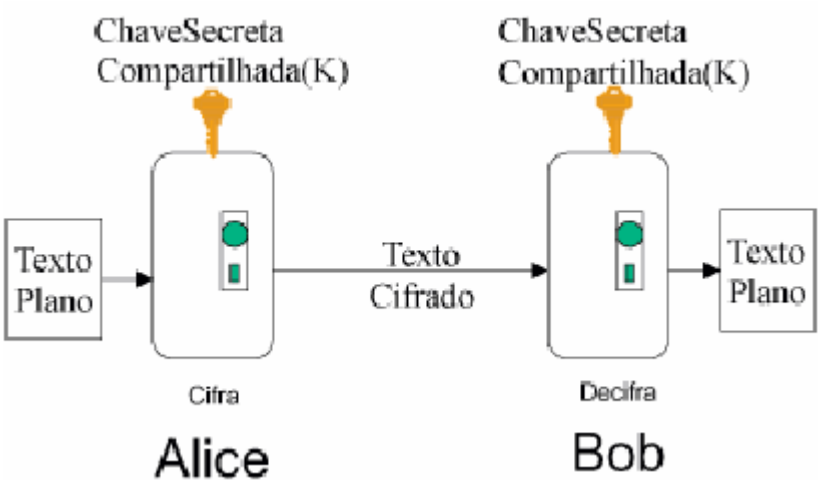

Figura 4.1 – Modelo simplificado de criptografia de chave privada, onde Alice utiliza chave K para cifrar o texto. E Bob através da mesma chave K, decifra o texto cifrado.

Fonte: [MISAGHI, 2001].

4.2. Criptografia de Chave Pública

A criptografia de chave pública que também é conhecida como criptografia assimétrica, utiliza duas chaves, uma chave privada e outra chave pública, onde se utiliza uma chave para encriptar, e terá que usar a outra para decriptar. Ou seja, se usar a chave pública para cifrar, só decifrará com a chave privada, e vice-versa. Lembrando que a chave privada deve sempre ser mantida secreta, podendo a chave pública ser

conhecida normalmente. A figura 4.2 mostra um modelo de criptografia de chave pública.

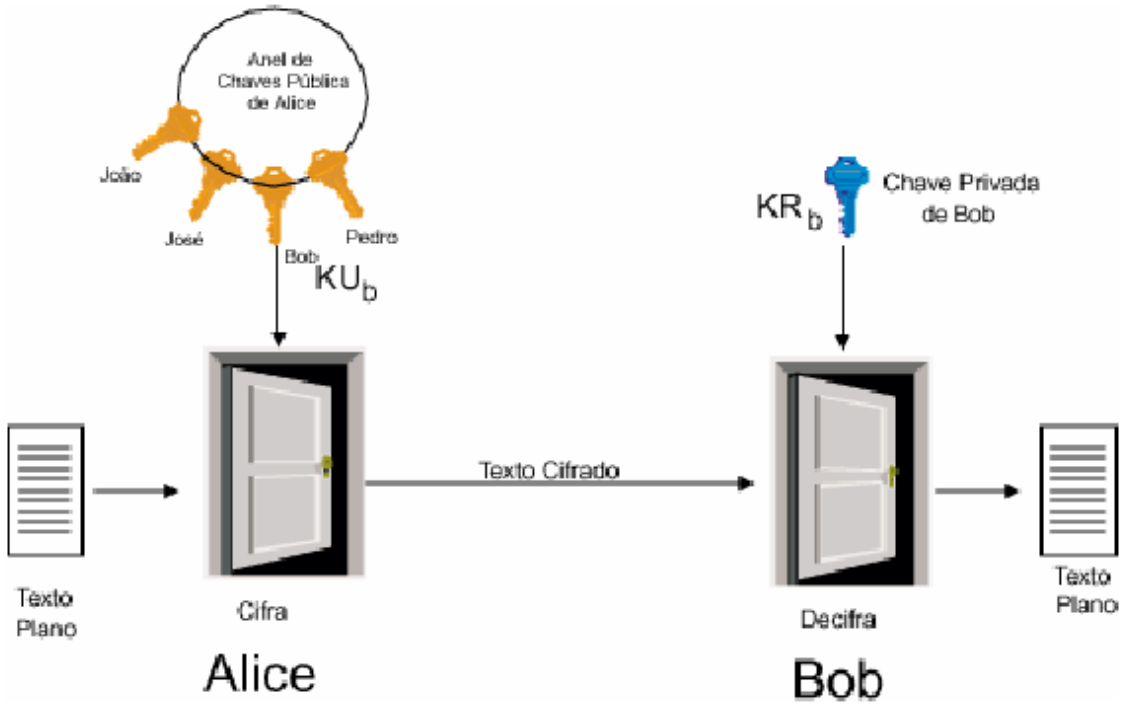

Figura 4.2 – modelo de criptografia de chave pública, onde Alice usa a chave pública do Bob, KUb, para cifrar a mensagem. E Bob, usa a chave privada, KRb para decifra-la.

Fonte: [MISAGHI, 2001].

4.3. Função de Hash

A função de hash também conhecida como resumos digitais, é uma espécie de assinatura ou impressão digital que representa o conteúdo de um fluxo de dados. Um hash que pode ser chamado de digesto pode ser

comparado com um selo de embalagem que indica clara e inequivocadamente se a embalagem já foi aberta ou violada.

Existe uma diferença entre hashs e cifragens, as cifragens transformam os dados do texto claro num criptograma e vice-versa, ou seja, é uma operação de duas vias, além disso, o tamanho do criptograma geralmente é igual ao comprimento do texto claro. Os hashs transformam os dados do texto claro ou cifrado num pequeno digesto, de tamanho fixo, numa operação de via única. Uma operação de via única não tem volta, ou seja, não é possível obter o texto claro a partir de um resultado hash.

A função de hash produz selos de segurança de comprimento fixo, não importando o comprimento do fluxo de dados ou do arquivo que representem, qualquer alteração efetuada no arquivo, por mínima que seja, altera substancialmente o resultado hash. Isto ocorre porque, mesmo se apenas um dos bits do arquivo for alterado, muitos bits do resultado serão afetados.

Os principais algoritmos de função de hash são:

- SNEFRU

É função hash de via única desenvolvida por Ralph Merkle que cria resultados hash de 128 ou de 256 bits.

- MD4

Desenvolvida por Ron Rivest que também produz hash de 128 bits.

- MD5

Também desenvolvida por Ron Rivest, é uma melhoria do MD4, que produz um hash de 128 bits.

- SHA

Desenvolvido pela NSA, produz hash de 160 bits.

4.3.1. MD5

O MD5 (Message Digest 5), é um algoritmo de hash de 128 bits unidirecional, que pelo meio de funções matemáticas, mapeiam uma cadeia de bits de qualquer tamanho em um número determinado de bits, que é o que chamamos de hash, conhecido também como resumo digital. Foi desenvolvido por Ron Rivest em 1991, e é muito usado quando se precisa comprimir textos longos de forma segura antes de ser cifrado com uma chave privada. Por ser um algoritmo unidirecional, uma hash MD5 não pode ser transformada novamente no texto que lhe deu origem, ou seja, dada uma entrada ele computa facilmente o valor do hash, mas o cálculo no sentido oposto é muito difícil ou mesmo impossível. O método de verificação é feito através da comparação das duas hashs, a da base de dados, e a outra da tentativa de login. O MD5 foi projetado para máquinas de 32 bits, podendo ser facilmente programado de forma compacta.

5. PROTÓTIPO

5.1. Ferramenta utilizada

A implementação do projeto foi realizada com a utilização do *software* MATLAB, versão 6.5, *Release* 13.

O MATLAB é uma linguagem de programação relativamente de fácil utilização, possuindo um conjunto de ferramentas para as mais diversas aplicações, apresentando ainda um ambiente rico para a visualização de dados, por ser dono de uma poderosa capacidade gráfica.

5.2. Métodos e recursos utilizados

Para o desenvolvimento da implementação do projeto, foram utilizados os seguintes métodos e recursos:

- Esteganografia;
- Criptografia - Função de Hash MD5 – Mesage Digest 5;
- LSB – *Least Significant Bit;*

5.3. Interface gráfica

A interface gráfica do protótipo foi totalmente construída explorando os recursos *Handle Graphics* do MATLAB, a fim de criar um ambiente funcional de fácil operação, além de apresentar os resultados obtidos de uma forma bem clara.

5.4. Processos

5.4.1. Ocultação do Texto na Imagem

O processo de ocultação do texto na imagem JPEG pode ser descrito simplificadamente da seguinte forma:

- 1° passo: Escolha uma imagem JPEG RGB de 24 *bits* a ser ocultada;
- 2° passo: Digite o texto a ser ocultado;
- 3° passo: Digite uma senha;
- 4° passo: A senha é passada pela função MD5 que irá gerar um hash de 128 bits;
- 5° passo: Os bits do hash serão passados por uma função lógica XOR juntamente com os bits do texto, gerando assim o texto criptografado;
- 6° passo: Os bits do texto criptografado serão inseridos nos LSBs de cada pixel da imagem, dessa forma, ocultando o texto na imagem e gerando uma imagem esteganografada.

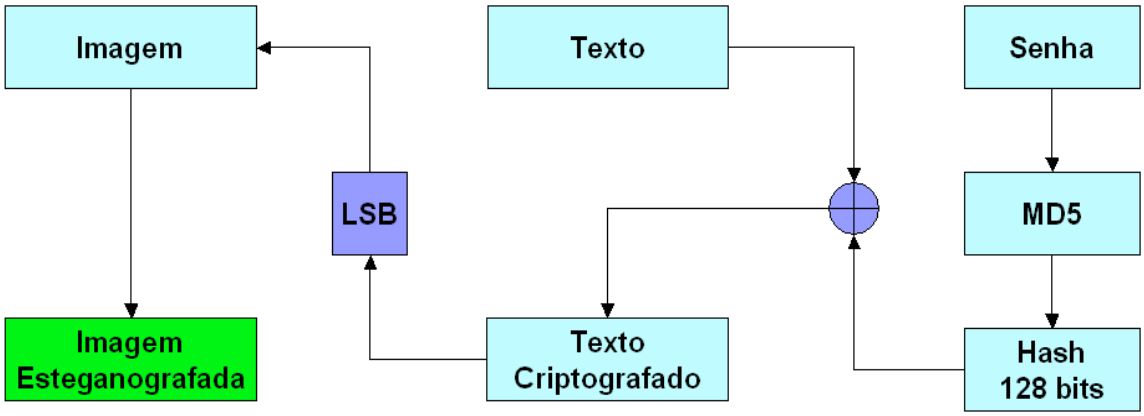

Figura 5.1 – fluxograma da ocultação da imagem

5.4.1.1. Verificação da imagem esteganografada

Ainda dentro do processo de ocultação, o aplicativo disponibiliza uma oportunidade de fazer uma verificação da imagem esteganografada, onde é feita uma comparação entre a imagem original e a imagem estego imagem a fim de verificar se ouve perda de qualidade ou degradação da imagem após ter sido ocultado o texto, e é basicamente assim:

- 1° passo: Mostra os pixels alterados da imagem esteganografa em relação a imagem original;
- 2° passo: Histograma das imagens original e esteganografada;
- 3° passo: Mostra dados estatísticos como média, desvio padrão e coeficiente de relação.

5.4.2. Extração do texto

O processo de extração do texto na imagem esteganografada é descrito da seguinte forma:

- 1° passo: Escolha a imagem esteganografada;
- 2° passo: É extraído o texto criptografado dos LSBs da imagem;
- 3° passo: Digite a mesma senha utilizada no processo de ocultação;
- 4° passo: A senha é passada pela função MD5, gerando um hash;
- 5° passo: Os bits do hash serão passados na função XOR juntamente com os bits do texto criptografado, dessa forma, gerando o texto original.

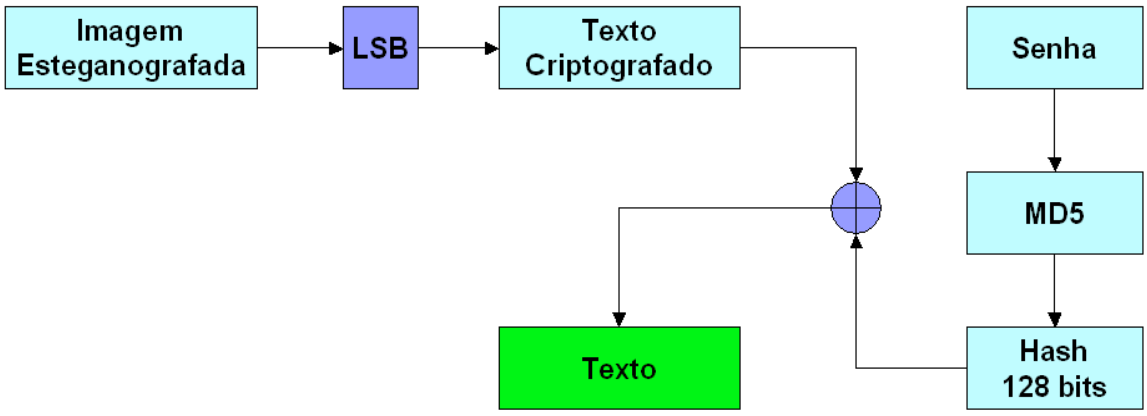

Figura 5.2 – fluxograma da extração do texto

5.5. Telas do protótipo

Chamando a função principal *stegjpeg* do protótipo, através da janela *Command* do MATLAB, surge uma tela com três opções, mostrada na figura 5.3 Cada uma dessas opções altera a tela principal, de forma a surgirem outras opções.

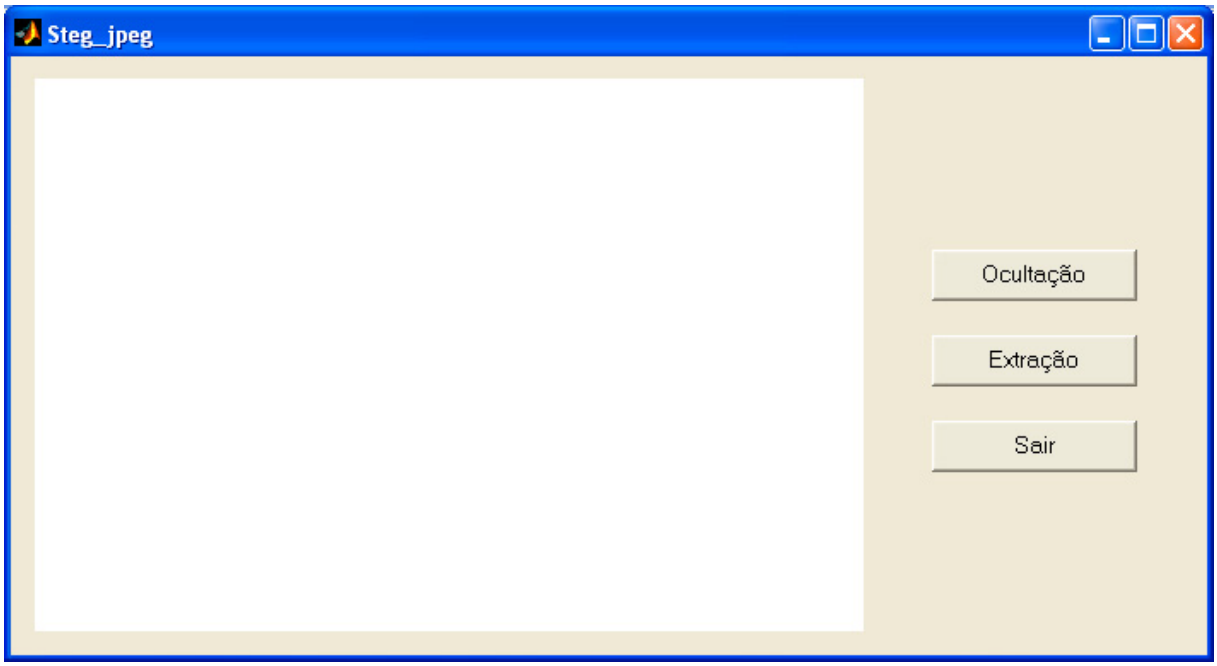

Figura 5.3 – tela principal

Clicando na opção ocultação aparece uma nova tela com novas opções como mostra a figura 5.4, essa tela possibilita a escolha da imagem a ser esteganografada, digitar o texto a ser ocultado, ocultar o texto e criptografar o mesmo através de uma senha, além de poder verificar a qualidade da imagem esteganografada comparando-a com a imagem original.

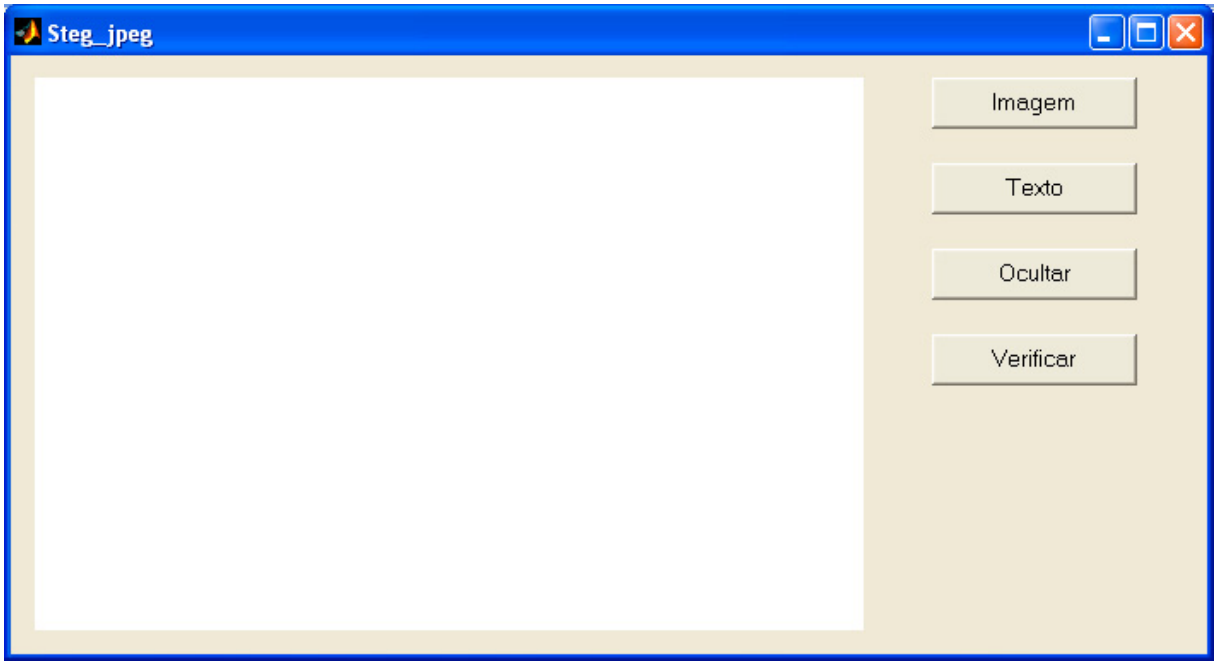

Figura 5.4 – Tela do processo de ocultação do texto.

A figura 5.5 mostra a escolha da imagem a ser esteganografada, o que acontece após clicar na opção imagem.

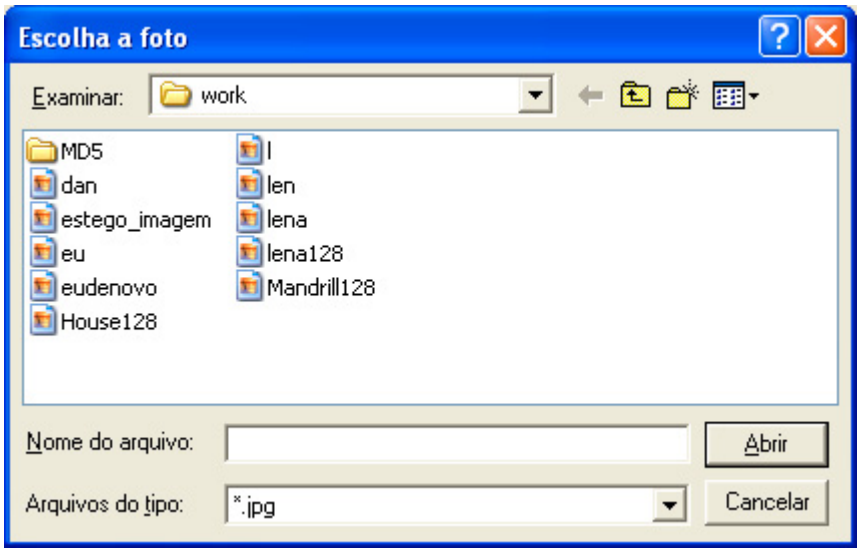

Figura 5.5 – Escolha da imagem a ser esteganografada

Na figura 5.6, pode-se ver a imagem escolhida.

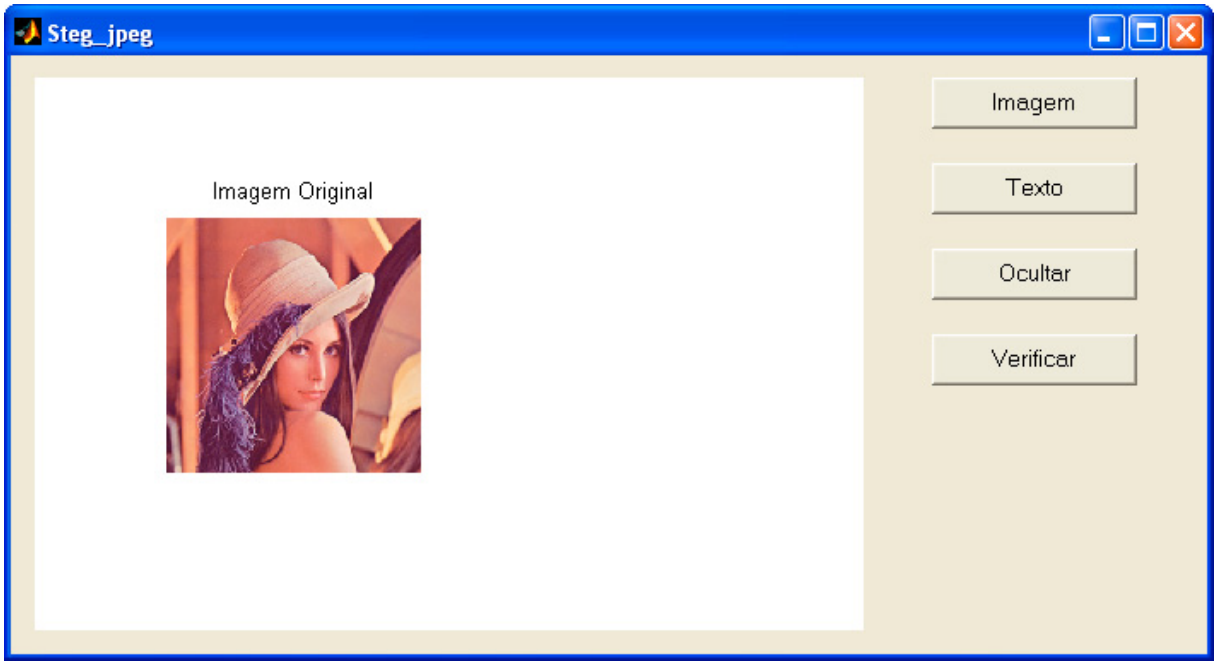

Figura 5.6 – Imagem escolhida para ser esteganografada

Clicando na opção texto surgirá a tela onde poderá ser digitado o texto, mostrado na figura abaixo.

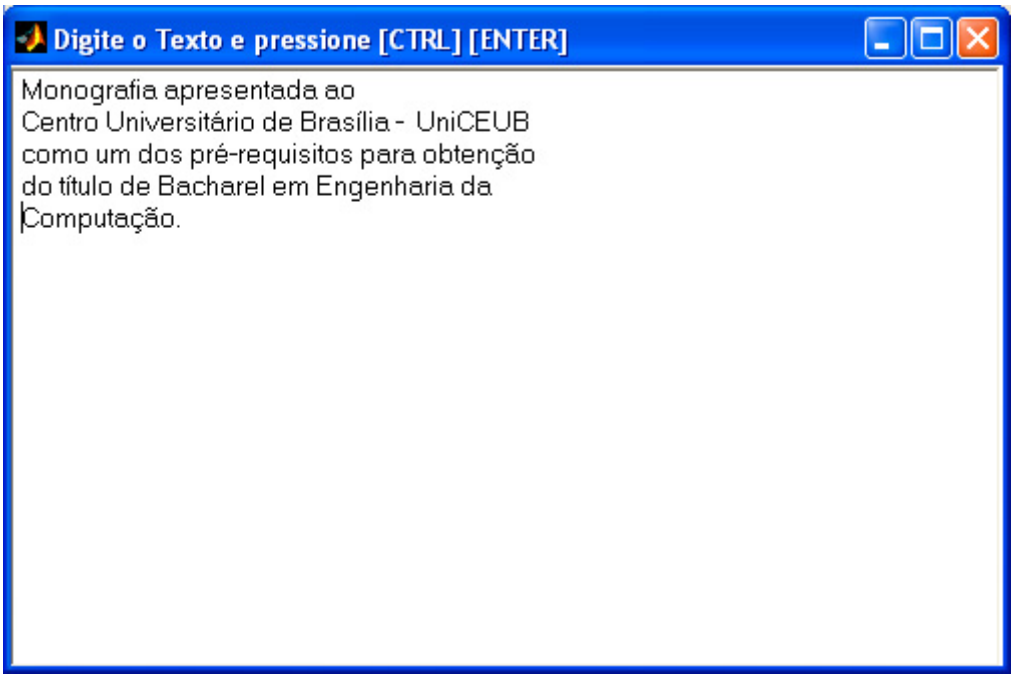

Figura 5.7 – Digitando o texto

Já a opção ocultar irá solicitar uma senha, o que mostra a figura abaixo.

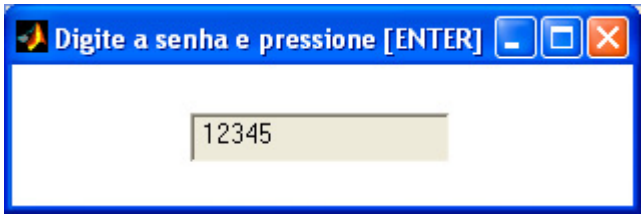

Figura 5.8 – Digitando a senha

Após digitar a senha surgirá uma nova tela para que se escolha o nome da imagem onde foi ocultado o texto. Tendo que obter a extensão.jpg, conforme mostra a figura 5.9.

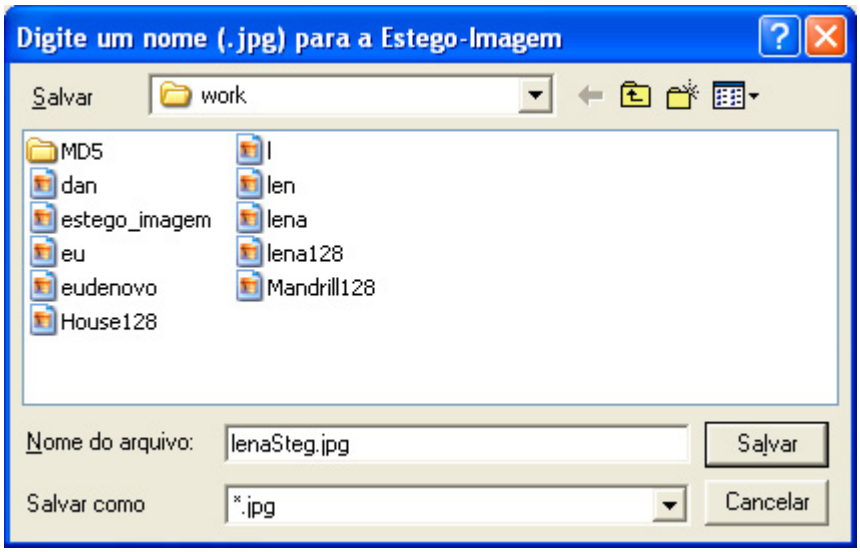

Figura 5.9 – Salvando a imagem esteganografada

Após salvar a imagem que foi ocultada o texto, o aplicativo mostra as duas imagens lado a lado (conforme a figura 5.10). E a mesma está

pronta para ser transmitida com a segurança do texto que foi ocultado garantida.

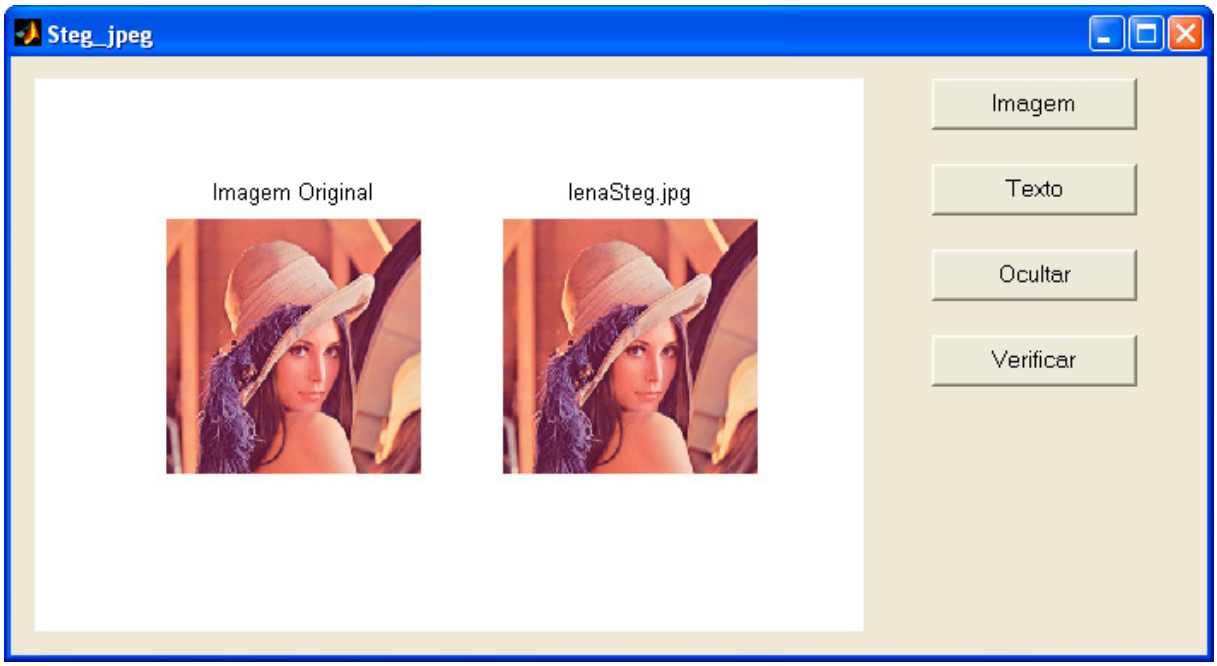

Figura 5.10 – Imagem original e imagem esteganografada

O próximo passo então seria a verificação da imagem esteganografada comparando-a com a imagem original, clicando na opção verificar o aplicativo oferece algumas opções para essa verificação, como mostra a figura 5.11.

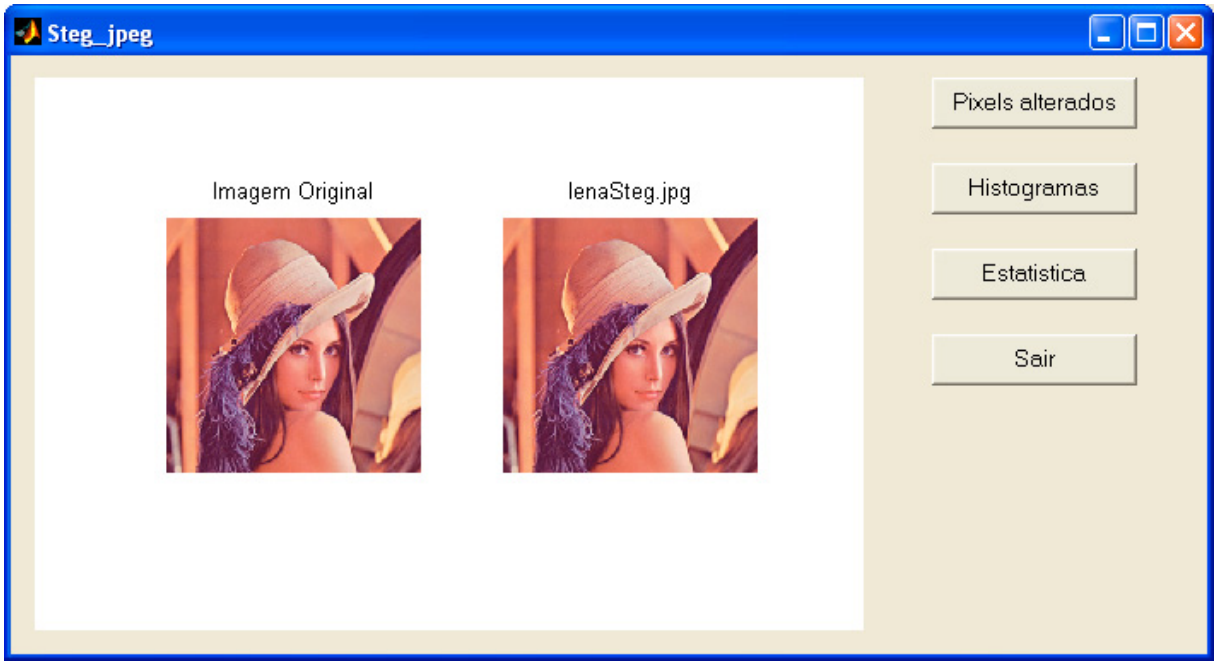

Figura 5.11 – Tela de verificação

A figura abaixo mostra os pixels alterados na imagem após ter sido ocultado o texto.

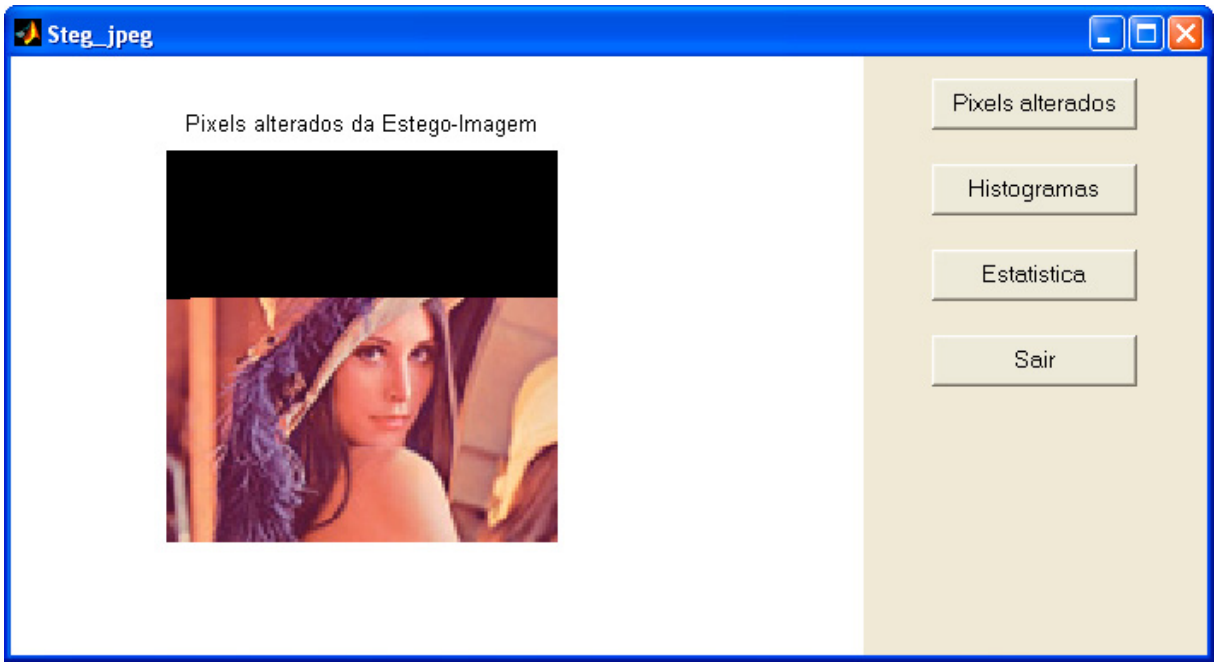

Figura 5.12 – Pixels ocupados pelo texto na imagem esteganografada

A figura 5.13 mostra um histograma das duas imagens, original e esteganografada.

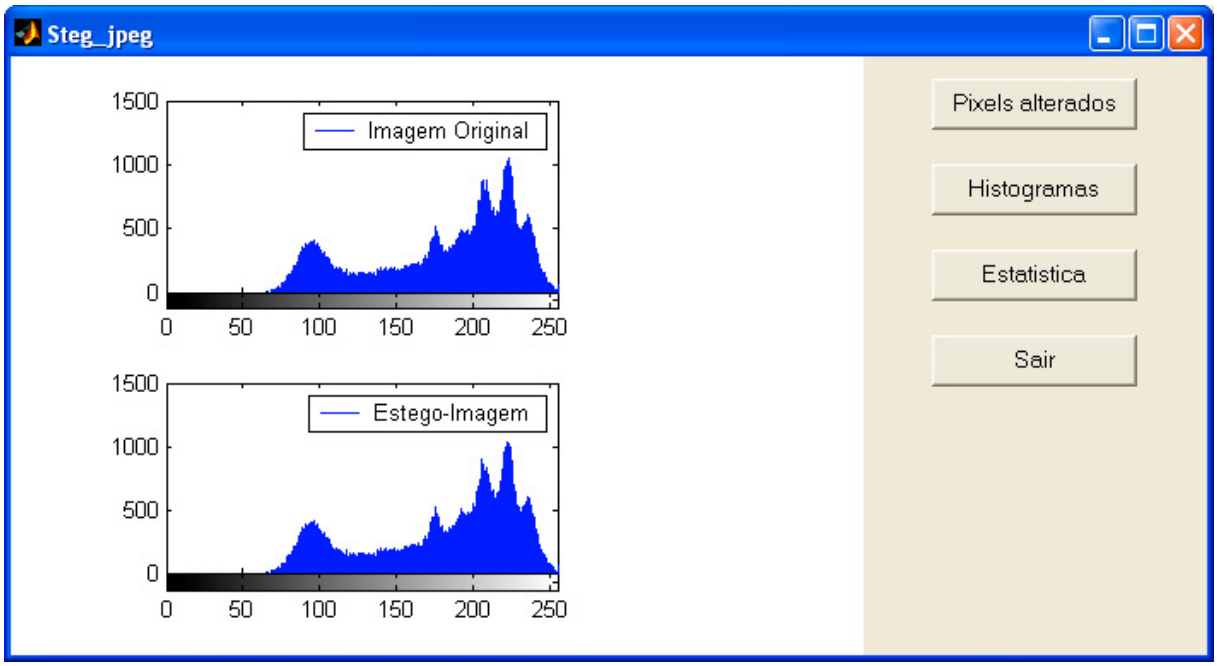

Figura 5.13 – Histograma da imagem original e esteganografada

Por ultimo o aplicativo mostra alguns dados estatísticos entre as duas imagens. Conforme a figura 5.14.

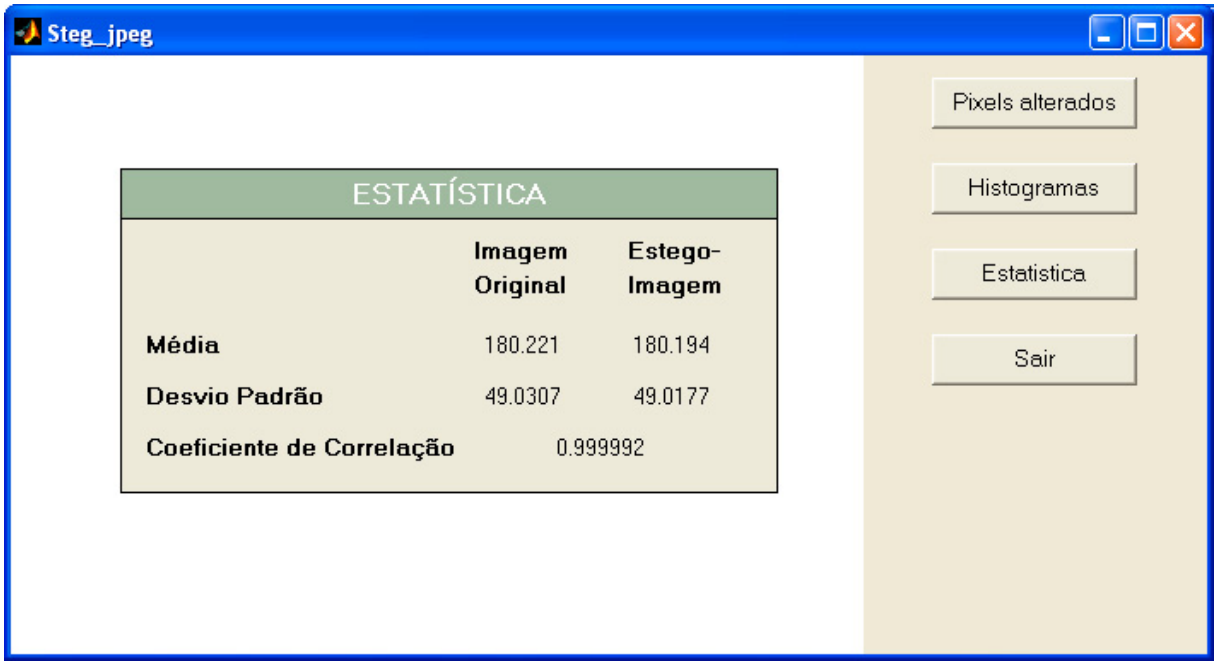

Figura 5.14 – Dados estatísticos

Ao clicar na opção sair ele fecha o aplicativo.

Na tela principal mostrada na figura 5.4 clicando na opção extração aparecerá três opções, uma para escolher a imagem da qual se deseja extrair o texto, uma para extrair o texto e outra para sair, como mostra a figura 5.15.

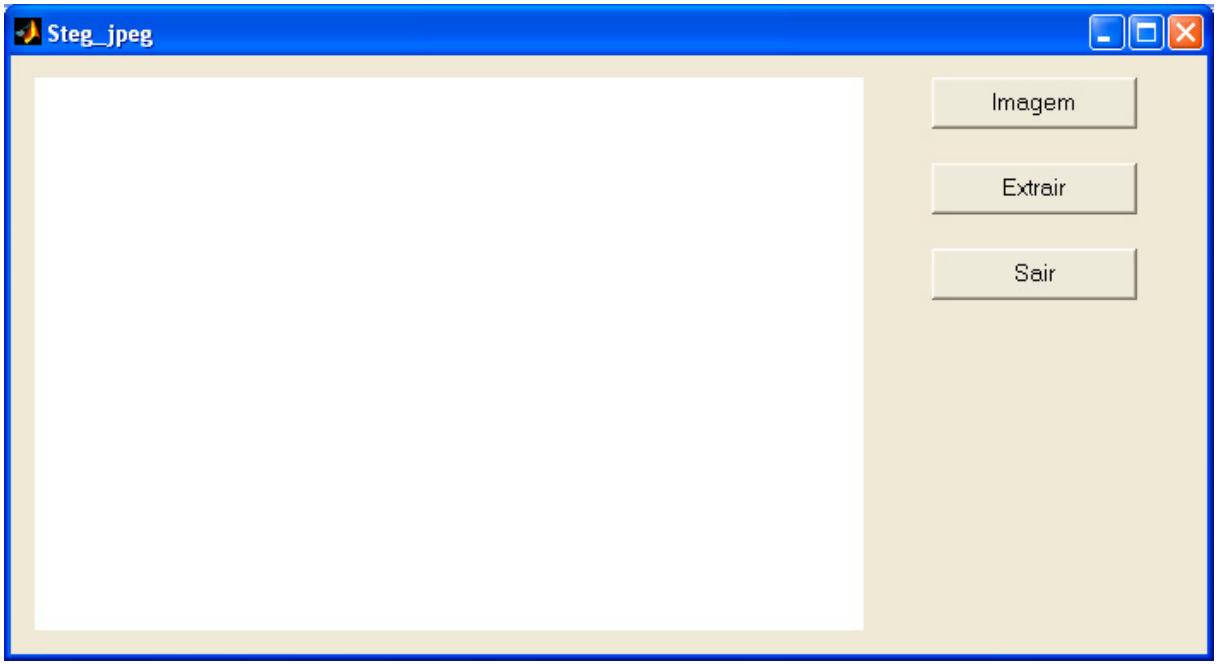

Figura 5.15 – Tela do processo de extração do texto.

Clicando na opção imagem, uma nova tela é aberta, onde poderá escolher a imagem esteganografada para extrair o texto.

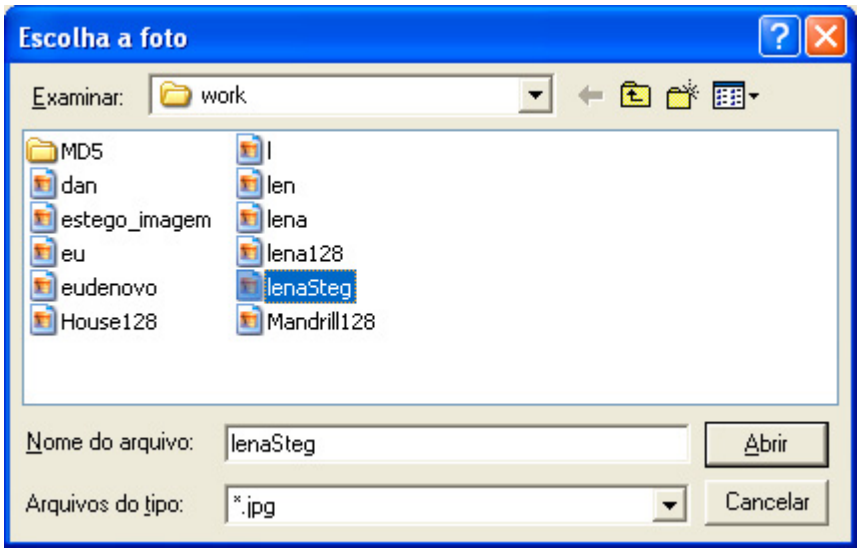

Figura 5.16 – Escolha da imagem da qual deseja extrair o texto

Após selecionar a imagem e abrir ela aparecerá na tela do aplicativo.

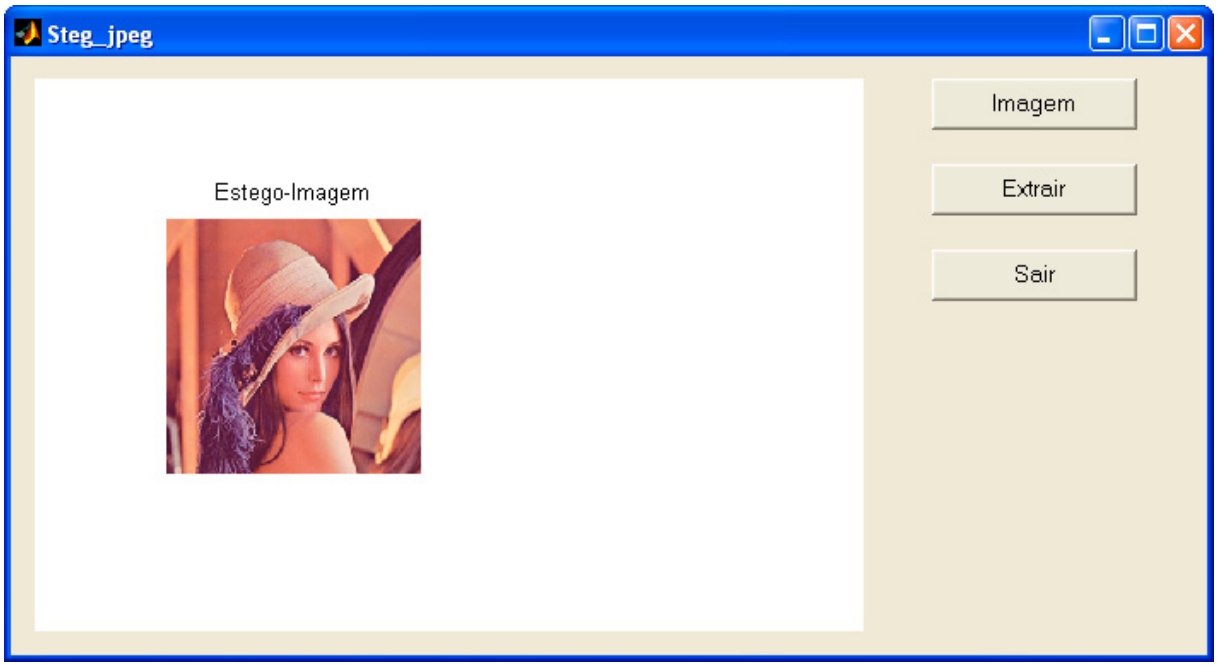

Figura 5.16 – Imagem escolhida

Clicando na opção extrair será solicitado uma senha, o importante é que por estar usando no desenvolvimento o recurso criptográfico MD5, e pelo mesmo se tratar de uma função de Hash unidirecional, será necessário utilizar a mesma senha que foi usada para ocultar, para extrair o texto, caso contrário o texto será totalmente incompreensível.

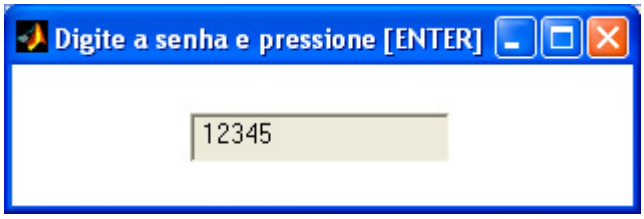

Figura 5.17 – Digitando a senha para extração do texto

Após ter digitado a senha, mostrará o texto extraído da imagem.

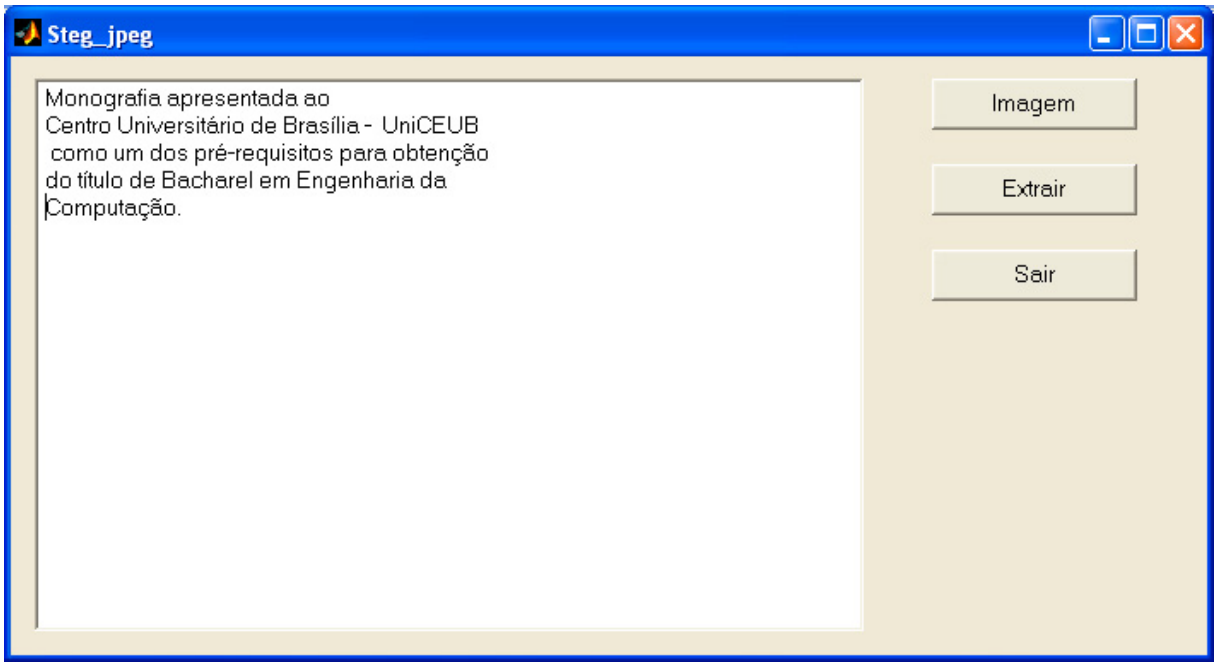

Figura 5.18 – Texto Extraído

6. TESTES E SIMULAÇÕES

Para os testes e simulações foram utilizadas algumas imagens da USC – *University of Southern Califórnia,* que são encontradas na internet, e também imagens próprias como fotos pessoais. Foram utilizados vários textos digitados na própria tela do aplicativo ou texto já prontos no formato Word, como é o caso do texto que aparece nos testes e simulações apresentados neste capítulo. Os testes e simulações foram divididos em duas partes, verificação da imagem esteganografada e extração do texto. Por serem considerados pontos relevantes no protótipo, um por direcionar a qualidade da imagem após a ocultação do texto e o outro por garantir que o texto após ser ocultado será o mesmo ao ser extraído, testando assim a segurança do protótipo.

6.1. Verificação da imagem esteganografada

Após a ocultação do texto criptografado na imagem, é possível verificar a qualidade da imagem esteganografada, comparando-a com a imagem original, ao clicar no botão verificar, é apresentado as duas imagens lado a lado na tela, a imagem original e a esteganografada, este botão ainda oferece opções para outras verificações conforme mostra a figura abaixo.

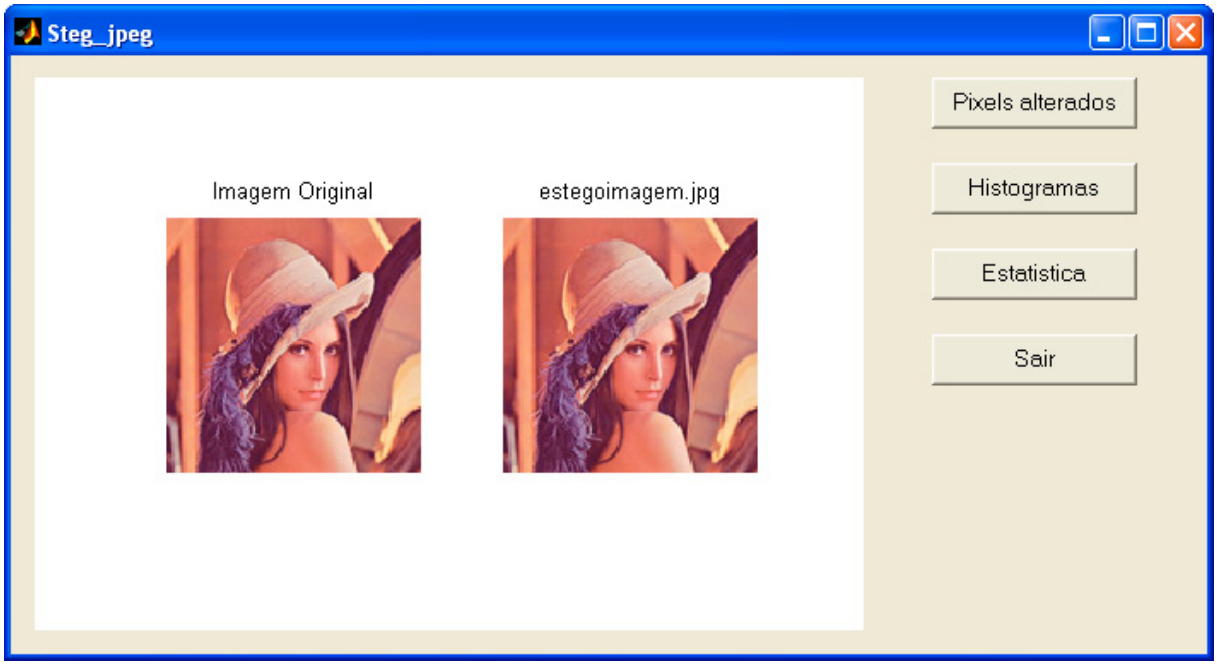

Figura 6.1 - Verificação

A imagem utilizada possui 16.384 pixels e 25,4 kb, e o texto 792 caracteres no formato Word, com trechos de um texto de W. Shakespeare.

6.1.1. Histograma e Dados estatísticos

Para uma melhor verificação da imagem esteganografada o aplicativo apresenta um histograma a fim de possibilitar uma comparação entre a imagem original e a imagem esteganografada, e apresenta também dados estatísticos. As figuras abaixo mostram o histograma e os dados estatísticos no plano RED, das imagens da figura 6.1.

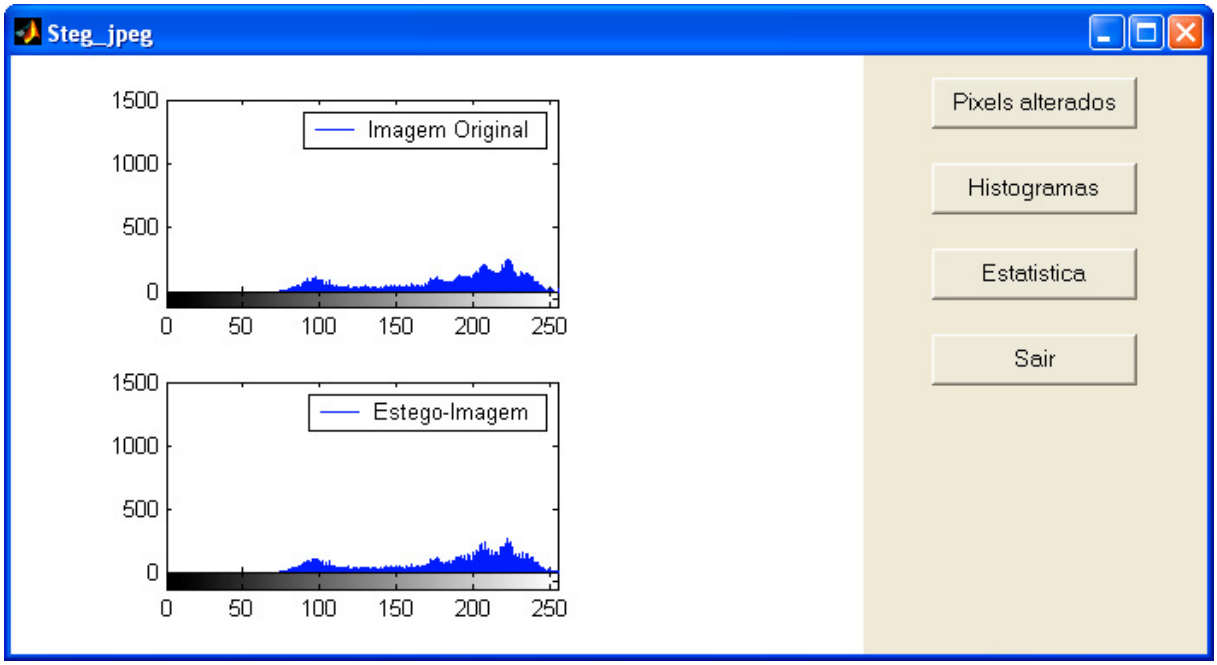

Figura 6.2 – Histograma do plano RED das imagens da figura 6.1

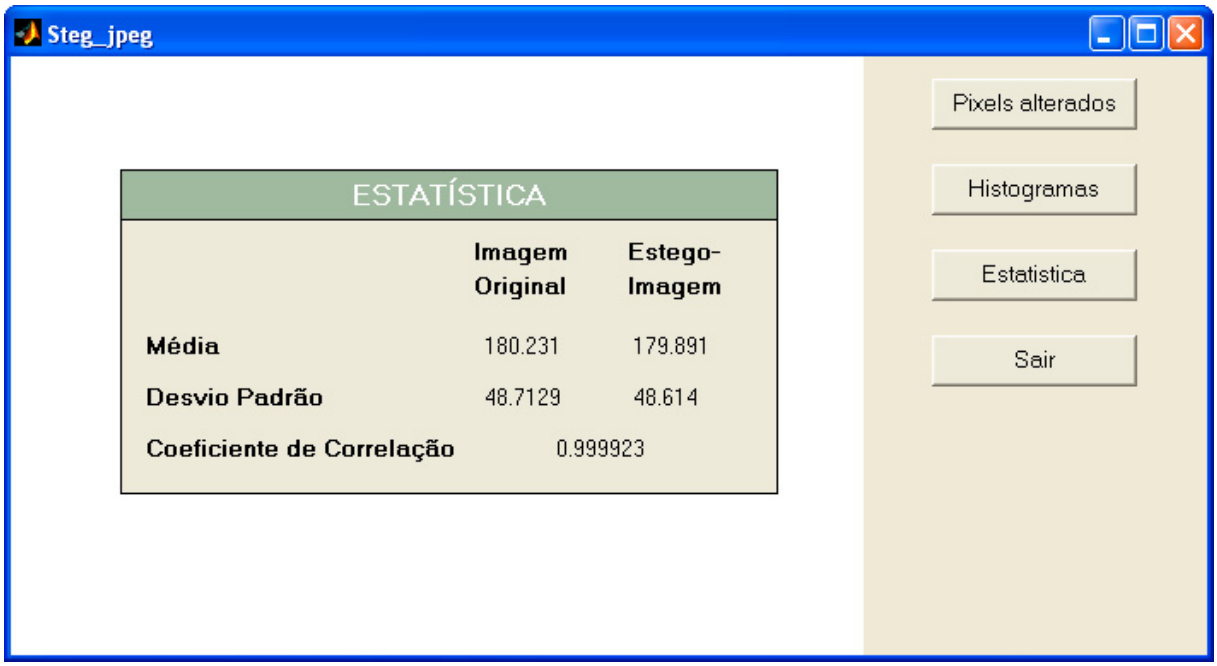

Figura 6.3 – Dados estatísticos

Os resultados apresentados acima mostram que tanto o histograma quanto os dados estatísticos (média e desvio padrão) das duas imagens comparadas, são bem semelhantes. Outro dado importante é o coeficiente

de correlação, entre a imagem original e a imagem marcada, que apresentou valores em próximo de 1, assim constatando a semelhança entre as duas imagens.

A figura 6.4 mostra o teste realizado com outra imagem, que possui 101.376 pixels, e o foi ocultado mesmo texto.

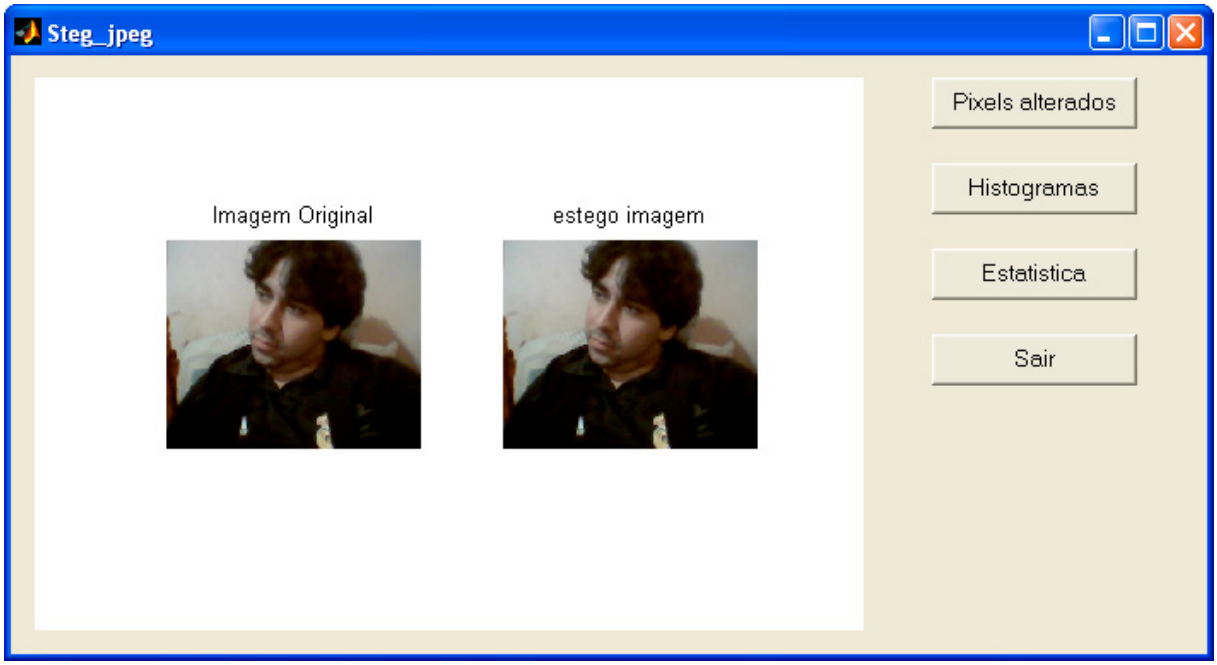

Figura 6.4 - Verificação

As figuras 6.5 e 6.6 mostram o histograma e os dados estatísticos no plano RED da imagem original e imagem esteganografada.

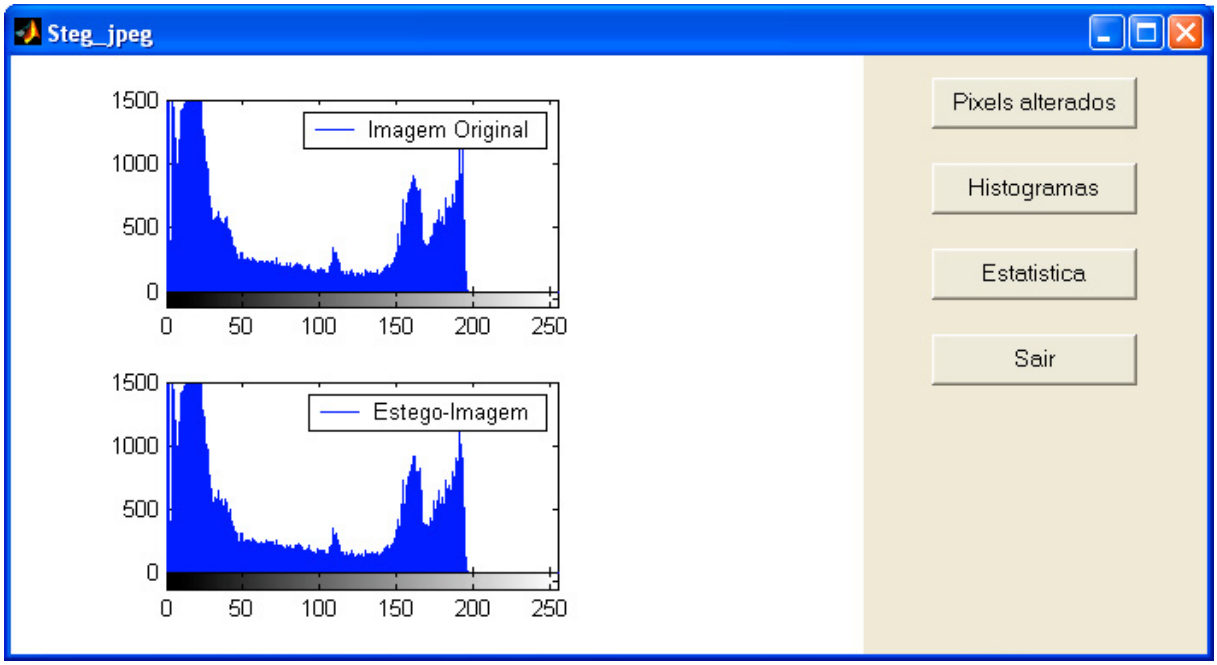

Figura 6.5 – Histograma do plano Red das imagens da figura 6.4

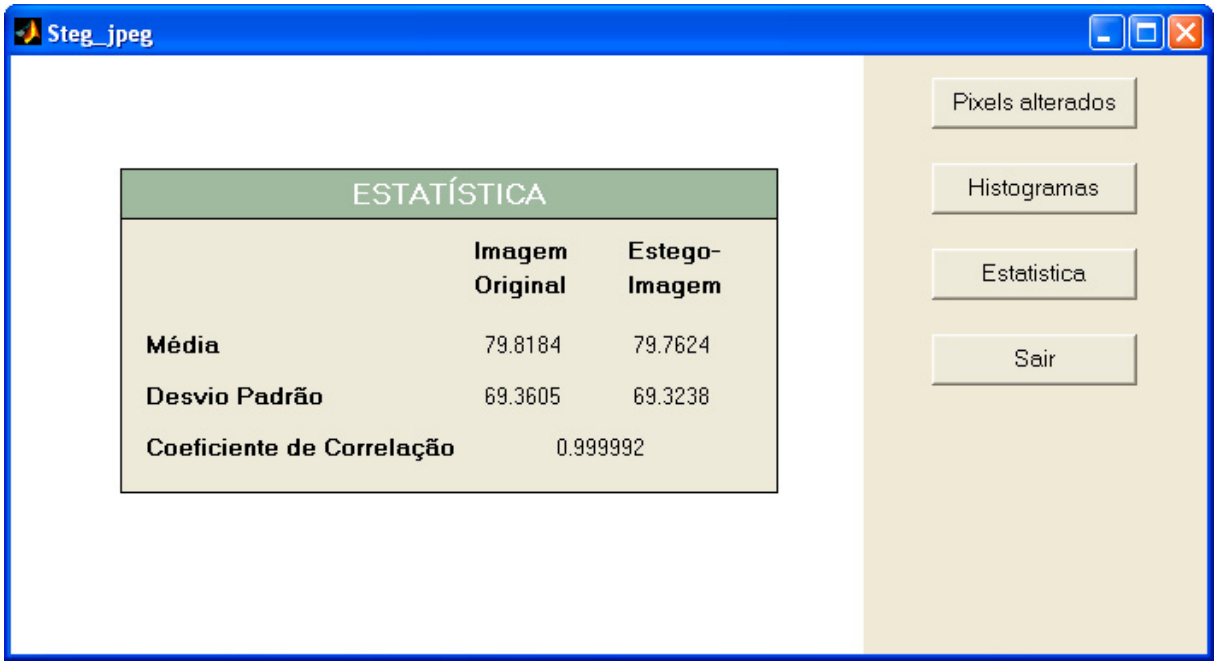

Figura 6.6 – Dados estatísticos

Os resultados mostram que o histograma e os dados estatísticos são bem semelhantes, assim como os resultados das outras imagens testadas.

6.2. Verificação do Texto extraído

Foram realizados testes e simulações para verificar se o texto ocultado estava sendo extraído com sucesso, principalmente por estar sendo utilizado no projeto uma função de hash unidirecional de 128 bits, que é o MD5, com isso a senha usada para ocultar tem que ser a mesma para extrair o texto, então foi testada também a segurança dessa senha, ou seja, se após ocultar o texto, ao extrair o texto não tinha sofrido nenhuma alteração. Os testes foram divididos em dois, um com a senha correta e outra com a senha incorreta.

6.2.1. Senha Correta

Para ocultação do texto estamos utilizando a mesma imagem e texto mostrados nos testes de verificação da imagem esteganografada.

A figuras abaixo mostram o texto a ser ocultado e senha utilizada para ocultar o mesmo.

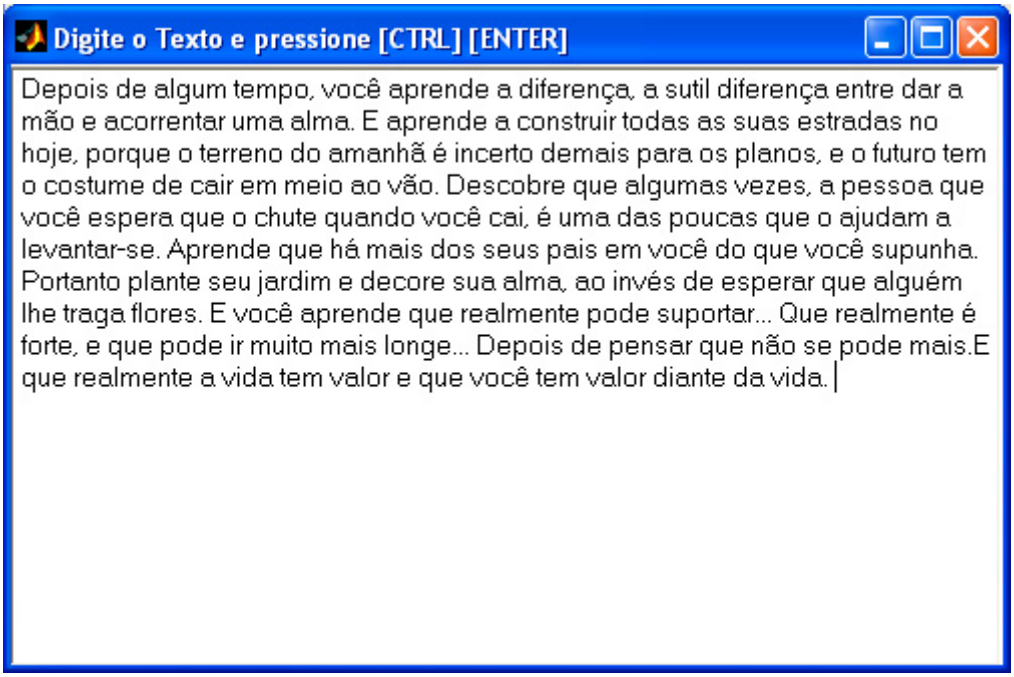

Figura 6.7 – Texto a ser ocultado

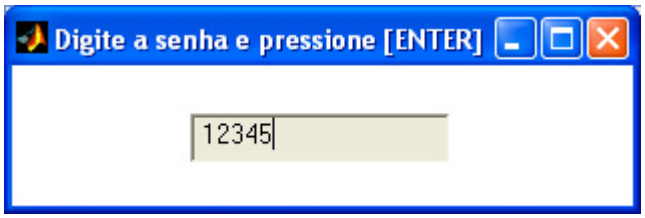

Figura 6.8 – Digitando a senha para ocultação

Para extrair o texto com sucesso será necessário utilizar a mesma senha, após ter sido escolhida a imagem esteganografada o aplicativo irá solicitar a senha como mostra a figura 6.9.

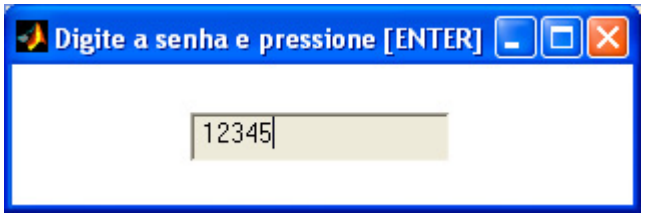

Figura 6.9 – Digitando a senha correta

A figura 6.10 mostra o texto após ter sido digitada a mesma senha utilizada para ocultação.

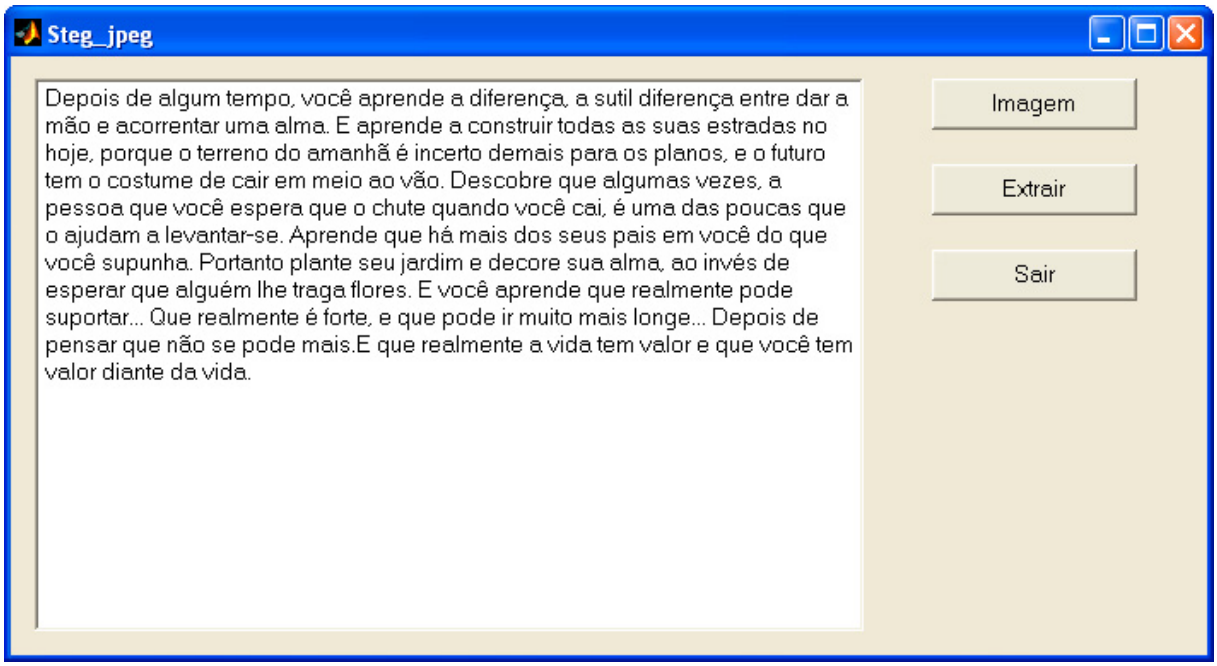

Figura 6.10 – Texto extraído com sucesso

Os resultados apresentados nas figuras acima comprovam que o texto extraído, além de ter sido extraído com sucesso, é o mesmo texto que foi ocultado, alcançando assim o objetivo do projeto.

6.2.2. Senha Incorreta

Caso a senha digitada para extrair o texto seja diferente da senha digitada para ocultar o texto, este será incompreensível. Conforme será mostrado a seguir.

Foi ocultado o mesmo texto na mesma imagem utilizando a mesma senha, mas para extrair foi utilizada outra senha, conforme a figura abaixo.

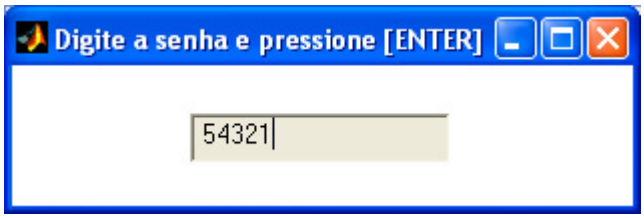

Figura 6.11 – Digitando senha incorreta

A figura 6.12 mostra o texto extraído após ter digitado uma senha incorreta, ou seja, uma senha diferente da que foi usada para ocultar.

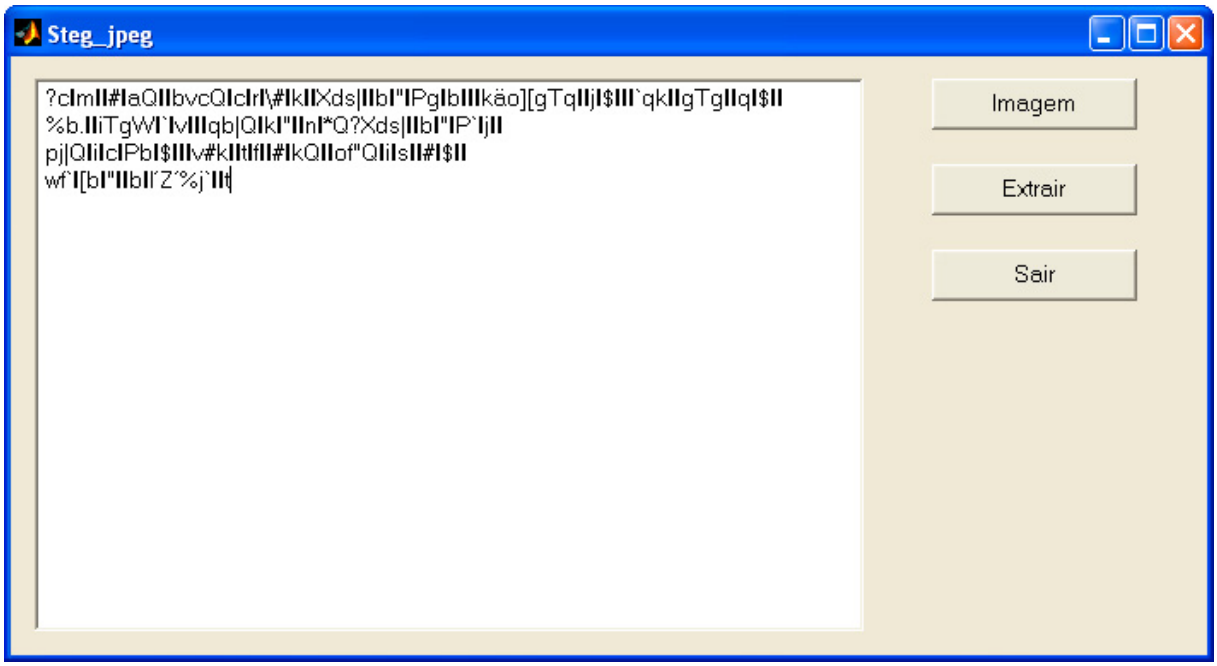

Figura 6.12 – Texto extraído

Com isso conclui-se que a segurança do texto ocultado está garantida, pois só será extraído o texto original se for digitado a mesma senha que foi utilizado para ocultar o texto, e por se tratar de uma função de hash unidirecional de 128 bits, fica extremamente difícil quebrar essa chave criptográfica garantindo assim a segurança do texto ocultado.

7. CONCLUSÃO

Com a realização projeto, conclui-se que é possível e foi realizada com sucesso a implementação de um aplicativo que oculte um texto em uma imagem no formato JPEG, e em seguida extraia o mesmo, utilizando a técnica de esteganografia em conjunto com a criptografia, desenvolvido totalmente no ambiente e linguagem MatLab, inclusive apresentando uma interface gráfica com botões para cada uma das ações.

Além de poder também verificar a integridade e qualidade da imagem após de inserir um texto criptografado na mesma, certificando que a imagem não sofre qualquer perda.

Por fim, pode-se destacar que o desenvolvimento do projeto foi bem sucedido, pois, os objetivos foram alcançados satisfatoriamente.

7.1. Trabalhos futuros

Nas pesquisas para o desenvolvimento deste trabalho, surgiram seguintes sugestões para trabalhos futuros:

- Implementar o protótipo em outra linguagem de desenvolvimento (C, Delphi, Java, etc);
- Implementar utilizando outro formato de arquivo, como áudio, ou vídeo;
- Utilizar outros recursos criptográficos, a fim de aumentar a segurança.
- Implementar utilizando a esteganografia TCP/IP;

8. REFERÊNCIAS BIBLIOGRÁFICAS

BUCHMANN, Johannes A. *Introdução à Criptografia*. São Paulo, 2002.

BENDER, Walter *et al*. *Technques for data hiding. In IBM Systens Jounal, 1996*. Disponível em: http://www.research.ibm.com/journal/sj/mit/sectiona/bender.html. Último acesso em 15/10/2006.

CASACURTA, Alexandre. *Computação Gráfica*. Disponível em: http://www.inf.unisinos.br/~osorio/CG-doc/cg.pdf. Último acesso em 16/09/2006.

CHAPMAN, Stephen J. *Programação em MATLAB para Engenheiros*. São Paulo: Pioneira Thomson Learning, 2003.

FALCÃO, Alexandre Xavier. *Fundamentos de Processamento de Imagem Digital.* Campinas: UNICAMP, 2003.

GOIS, Matheus C. A. Mascaramento de informações. *Histórico, definições e aplicações. Sociedade Brasileira de Computação,* 2003. Disponível em: http://www.sbc.org.br. Último Acesso em: 10/10/2006.

HANSELMAN, Duane; LITTLEFIELD, Bruce. *MATLAB 6. Curso Completo*. São Paulo: Prentice Hall, 2003.

JASCONE, Fábio Luiz Tavares. *Protótipo de Software para Ocultar texto Criptografado em Imagens Digitais*. Blumenau, 2003. Trabalho de Conclusão de Curso – Ciências da Computação, Universidade Regional de Blumenau.

MISAGHI, Mehran. *Avaliação de Modificações do Cifrador Caótico de Roskin*. Florianópolis, 2001. Dissertação (Mestrado em Ciência da Computação) – Centro Tecnológico, Universidade Federal de Santa Catarina.

Official JPEG homepage. Disponível em: http://www.jpeg.org. Último acesso em 10/09/2006.

OLIVEIRA, Wilson José. Segurança da informação – *Técnicas e Soluções*. Florianópolis, 2001.

OWENS, Mark. *A Discussion of Covert Channels and Steganography*, 2002. Disponível em: http://www.gray-world.net/papers/adiscussionofcc. Último Acesso em: 17/10/2006.

PETITCOLAS, F. A. Et al. Information hiding – a survey. *In Proceedings of IEEE*. Special issue on Protection of IEEE. Special issue on Protection on multimedia content, 1999.

PINHEIRO, José M. S. *Esteganografia Digital,* 2003. Disponível em http://www.projetoderedes.com.br. Último Acesso em: 08/10/2006.

ROCHA, Anderson de Rezende. *Camaleão. Um Software para Segurança Digital Utilizando Esteganografia*. Minas Gerais, 2003. Monografia (Ciência da Computação) – Departamento de Ciências da Computação, Universidade Federal de Lavras.

TRAINA, Agma Juci Machado; OLIVEIRA, Maria Cristina Ferreira de. *Apostila de Computação Gráfica*. São Paulo: Universidade de São Paulo, 2004. Disponível em: http://gbdi.icmc.usp.br/documentacao/apostilas. Último acesso em: 10/09/2006.

USC - University of Southern California. *The USC-SIPI Image Database*. Disponível em: http://sipi.usc.edu/database. Último acesso em: 20/09/2006.

APÊNDICE A – Função de Inicialização

Stegjpeg.m

```
% ------------------------------------------------------------------------
% Centro Universitario de Brasilia - UniCEUB
% Faculdade de Ciencias Exatas e Tecnologia - Faet
% Curso de Engenharia da Computaçao
% Disciplina: Projeto Final - 2º Semestre de 2006
% Prof. Orientador: Aderlon Queiroz
% Aluno: Daniel Ribeiro de Moura
% ------------------------------------------------------------------------
% Ocultaçao de texto criptografado em imagem de formato JPEG
% ------------------------------------------------------------------------
% Funçao stegjpeg - Inicializaçao do aplicativo
%
% Apaga a memoria e limpa a tela
%
clear all;
clc;
%
% Desenha a janela do programa
%
figure('Position',[150 195 700 350],...
    'Color','white',...
    'menubar','none',...
    'NumberTitle','off',...
    'Name','Steg_jpeg');
uicontrol('Style','frame',...
    'BackgroundColor',[0.93 0.91 0.83],...
    'ForegroundColor',[0.93 0.91 0.83],...
    'Position',[500 0 205 353]);
uicontrol('Style','frame',...
    'BackgroundColor',[0.93 0.91 0.83],...
    'ForegroundColor',[0.93 0.91 0.83],...
    'Position',[0 0 15 353]);
uicontrol('Style','frame',...
    'BackgroundColor',[0.93 0.91 0.83],...
    'ForegroundColor',[0.93 0.91 0.83],...
    'Position',[0 338 708 15]);
uicontrol('Style','frame',...
    'BackgroundColor',[0.93 0.91 0.83],...
    'ForegroundColor',[0.93 0.91 0.83],...
    'Position',[0 0 708 15]);
%
% Cria os botoes principais do programa
%
uicontrol('Style','pushbutton',...
    'String','Ocultação',...
    'Enable','on',...
    'Fontsize',10,...
    'Position',[540 208 120 30],...
    'callback','ocultar_callback');
uicontrol('Style','pushbutton',...
    'String','Extração',...
    'Enable','on',...
    'Fontsize',10,...
    'Position',[540 158 120 30],...
    'callback','extrair_callback');
uicontrol('Style','pushbutton',...
    'String','Sair',...
    'Enable','on',...
    'Fontsize',10,...
    'Position',[540 108 120 30],...
    'callback','sair_callback');
```

APÊNDICE B – Funções do Processo de Ocultação

Ocultar_callback.m

```
function ocultar_callback
    %
    % Cria os botoes do processo de ocultaçao do texto
    %
    uicontrol('Style','frame',...
        'BackgroundColor',[0.93 0.91 0.83],...
        'ForegroundColor',[0.93 0.91 0.83],...
        'Position',[500 0 205 353]);
    uicontrol('Style','pushbutton',...
        'String','Imagem',...
        'Enable','on',...
        'Fontsize',10,...
        'Position',[540 308 120 30],...
        'callback','imagem_callback');
    uicontrol('Style','pushbutton',...
        'String','Texto',...
        'Enable','on',...
        'Fontsize',10,...
        'Position',[540 258 120 30],...
        'callback','mensagem_callback');
    uicontrol('Style','pushbutton',...
        'String','Ocultar',...
        'Enable','on',...
        'Fontsize',10,...
        'Position',[540 208 120 30],...
        'callback','senha_callback');
    uicontrol('Style','pushbutton',...
        'String','Verificar',...
        'Enable','on',...
        'Fontsize',10,...
        'Position',[540 158 120 30],...
        'callback','verificar_callback');
```

imagem_callback.m

```
function imagem_callback
    global img_d
    global img
    global dsv
    %
    % Busca Imagem
    %
    [a_img, f_dir]=uigetfile('*.jpg', 'Escolha a foto');
    %
    % Le Imagem
    %
    img=imread([f_dir, a_img]);
    %
    % Converte a imagem em uma matriz numerica
    %
    img_d=double(img);
    %
    % Apresenta imagem escolhida na tela
    %
    subplot(1,3,1), imshow(img), title('Imagem Original');
    %
    % Atribui valor 1 para variavel dsv (desvio)
    % para que no processo de captaçao da senha seja
    % indicado que esta no processo de ocultaçao
    %
```

```
    dsv=1;
```

mensagem_callback.m

```
function mensagem_callback
    global txt
    %
    % Desenha a janela para receber o texto
    %
    figure('Position',[153 215 495 300],...
        'Color','white',...
        'menubar','none',...
        'NumberTitle','off',...
        'Name','Digite o Texto e pressione [CTRL] [ENTER]');
    uicontrol('Style','edit',...
        'Min',1,...
        'Max',18,...
        'BackgroundColor','white',...
        'position',[0 0 495 300],...
        'HorizontalAlignment','left',...
        'Fontsize',12,...
        'callback','edit_texto_callback');
```

edit_texto_callback.m

```
function edit_texto_callback
    global txt
    %
    % Pega o texto e atribui a variavel t
    %
    handle = gco;
    t = get(handle,'string');
    %
    % Concatena a letra ÿ que possui o valor em binario 11111111 com a variavel
    % t para que todas as letras do texto fiquem com a representaçao de 8 bits
    %
    i='ÿ';
    tx_s=strcat(i,t);
    %
    % Converte a string da variavel tx_s para uma representaçao ASCII e
    % atribui a variavel tx_d
    %
    tx_d=double(tx_s);
    %
    % Converte a variavel tx_d para binario e atribui a variavel tx_b
    %
    tx_b=dec2bin(tx_d);
    %
    % Pega a quantidade de bits do texto e acrescenta o valor no inicio e
    % atribui a variavel txt (string)
    %
    [n,m]=size(tx_b);
    v=(n*m)+32;
    vs=num2str(v);
    [o,p]=size(vs);
    c='0';
    cc='00';
    if p==2,
        vsc=strcat(cc,vs);
    elseif p==3,
        vsc=strcat(c,vs);
    elseif p==4,
        vsc=vs;
    end
    txt=strcat(vsc,tx_s);
    %
```

```
    % Fecha a janela de texto
    %
    close;
    drawnow;
```

ocultar.m

```
function ocultar
    global img
    global img_d
    global img_e
    global img_p
    global txt
    global hash
    %
    % Transforma a coluna da matriz hash em linha
    %
    hash2=hash';
    %
    % Converte a string da variavel txt para uma representaçao ASCII e
    % atribui a variavel msg
    %
    msg=uint8(txt);
    %
    % Pega o tamanho do texto
    %
    [i,j]=size(txt);
    %
    % Criptografa o texto
    %
    w=(j-5)/16;
    r=5;
    for x=1:w,
        for y=1:16
            msg(1,r+y)=bitxor(hash2(1,y),msg(1,r+y));
        end
        r=r+16;
    end
    msg_s=char(msg);
    msg_d=double(msg_s);
    msg_b=dec2bin(msg_d);
    [x,y]=size(msg_b);
    c=1;
    for i=1:x,
        for j=1:y,
            a(1,c)=msg_b(i,j);
            c=c+1;
        end
    end
    [V,T]=size(a);
    %
    % Pega o tamanho da Imagem
    %
    [M,N,O]=size(img_d);
    %
    % Zera os LSB's da Imagem
    %
    for z=1:O,
        for u=1:M,
            for v=1:N,
                if mod(img_d(u,v,z),2)==1,
                    img_0(u,v,z)=img_d(u,v,z)-1;
                else
                    img_0(u,v,z)=img_d(u,v,z);
                end
            end
        end
```

```
    end
    img_e=img_d;
    img_p=img_0;
    %
    % Insere o texto na imagem
    %
    c=1;
    Q=T+1;
    for u=1:M,
        for v=1:N,
            if c < Q & a(1,c)=='1',
                img_e(u,v,1)=img_0(u,v,1)-1;
                img_p(u,v,1)=0;
                img_p(u,v,2)=0;
                img_p(u,v,3)=0;
                c=c+1;
            elseif c < Q & a(1,c)=='0',
                img_e(u,v,1)=img_0(u,v,1);
                img_p(u,v,1)=0;
                img_p(u,v,2)=0;
                img_p(u,v,3)=0;
                c=c+1;
            elseif c > Q
                img_e(u,v,1)=img_d(u,v,1);
            end
        end
    end
    %
    % Salva a estego-imagem
    %
    [n_img, f_dir]=uiputfile('*.jpg', 'Digite um nome (.jpg) para a Estego-
Imagem');
    img_e=uint8(img_e);
    img_p=uint8(img_p);
    imwrite(img_e,n_img,'jpg','Mode','lossless');
    %
    % Apresenta imagem esteganografada na tela
    %
    subplot(1,3,2), imshow(img_e), title(n_img);
```

senha_callback.m

```
function senha_callback
    global dsv
    %
    % Desenha a janela para receber a senha
    %
    figure('Position',[153 446 310 70],...
        'Color','white',...
        'menubar','none',...
        'NumberTitle','off',...
        'Name','Digite a senha e pressione [ENTER]');
    uicontrol('Style','edit',...
        'position',[90 22 130 25],...
        'HorizontalAlignment','left',...
        'Fontsize',12,...
        'callback','edit_senha_callback');
```

edit_senha_callback.m

```
function edit_senha_callback
    global img_d
    global img
    global txt
    global dsv
    global hash
```

```
    %
    % Pega a senha digitada e atribui a variavel senha
    %
    handle = gco;
    senha = get(handle,'string');
    %
    % Converte a string da variavel senha
    % para numero inteiro sem sinal de 8 bits
    %
    senha=uint8(senha);
    %
    % Cria o arquivo senha.bin e coloca o conteudo da variavel senha para
    % gerar o hash (MD5)
    %
    sbin=fopen('senha.bin','wb');
    fwrite(sbin,senha,'int8');
    %
    % Fecha o arquivo senha.bin
    fclose(sbin);
    %
    % Calcula o hash de bloco.bin e coloca no arquivo hash.hsh
    %
    dos('c:\matlab6p5p1\work\md5\md5 -oc:\matlab6p5p1\work\hash.hsh
c:\matlab6p5p1\work\senha.bin');
    %
    % Abre o hash.hsh para leitura
    %
    fhsh=fopen('hash.hsh','rb');
    %
    % Separa apenas a string de hash (ASCII)
    %
    hash=fread(fhsh,16,'uint8');
    %
    % Fecha hash.hsh
    %
    fclose(fhsh);
    %
    % Fecha a janela da senha
    %
    close;
    drawnow;
    %
    % Desvia para o processo de ocultaçao ou extraçao de acordo com
    % o valor da variavel dsv
    %
    if dsv==1
       ocultar;
    else
       extrair;
    end
```

verificar_callback.m

```
function verificar_callback
    global img
    global img_d
    global img_e
    global img_p
    %
    % Cria os botoes do processo de verificaçao
    %
    uicontrol('Style','frame',...
        'BackgroundColor',[0.93 0.91 0.83],...
        'ForegroundColor',[0.93 0.91 0.83],...
        'Position',[500 0 205 353]);
    uicontrol('Style','pushbutton',...
        'String','Pixels alterados',...
```

```
        'Enable','on',...
        'Fontsize',10,...
        'Position',[540 308 120 30],...
        'callback','pixels_callback');
    uicontrol('Style','pushbutton',...
        'String','Histogramas',...
        'Enable','on',...
        'Fontsize',10,...
        'Position',[540 258 120 30],...
        'callback','histograma_callback');
    uicontrol('Style','pushbutton',...
        'String','Estatistica',...
        'Enable','on',...
        'Fontsize',10,...
        'Position',[540 208 120 30],...
        'callback','estatistica_callback');
    uicontrol('Style','pushbutton',...
        'String','Sair',...
        'Enable','on',...
        'Fontsize',10,...
        'Position',[540 158 120 30],...
        'callback','sair_callback');
```

pixels_callback.m

```
function pixels_callback
    global img_p
    % Apresenta na tela a imagem que destaca os pixels alterados
    clf;
    subplot(1,2,1), imshow(img_p), title('Pixels alterados da Estego-Imagem');
    verificar_callback;
```

histograma_callback.m

```
function histograma_callback
    global img
    global img_e
    %
    % Apresenta na tela os histogramas das imagens original e esteganografada
    %
    clf;
    subplot(2,2,1), imhist(img(:,:,1)), legend('Imagem Original');
    axis([0 255 0 1500]);
    subplot(2,2,3), imhist(img_e(:,:,1)), legend('Estego-Imagem');
    axis([0 255 0 1500]);
    verificar_callback;
```

estatistica_callback.m

```
function estatistica_callback
    global img
    global img_e
    %
    % Calcula a media das imagens original e estego-imagem
    %
    md_img=mean2(img(:,:,1));
    md_img_e=mean2(img_e(:,:,1));
    %
    % Calcula o desvio padrao das imagens original e estego-imagem
    %
    dv_ori=std2(img(:,:,1));
    dv_est=std2(img_e(:,:,1));
    %
    % Calcula o coeficiente de correlaçao entre as imagens
    % original e estego-imagem
```

```
%
crl_img_img_e=corr2(img(:,:,1),img_e(:,:,1));
%
% Desenha a tabela dos dados estatisticos
%
clf;
uicontrol('Style','frame',...
    'Position',[65 95 385 190]);
uicontrol('Style','frame',...
    'BackgroundColor',[0.64 0.73 0.62],...
    'ForegroundColor',[0 0 0],...
    'Position',[65 255 385 30]);
uicontrol('Style','text',...
    'String','ESTATÍSTICA',...
    'BackgroundColor',[0.64 0.73 0.62],...
    'ForegroundColor','white',...
    'FontWeight','bold',...
    'Fontsize',12,...
    'Position',[70 260 375 20]);
uicontrol('Style','text',...
    'String','Imagem',...
    'HorizontalAlignment','center',...
    'Fontsize',10,...
    'FontWeight','bold',...
    'Position',[250 225 100 20]);
uicontrol('Style','text',...
    'String','Original',...
    'HorizontalAlignment','center',...
    'Fontsize',10,...
    'FontWeight','bold',...
    'Position',[250 205 100 20]);
uicontrol('Style','text',...
    'String','Estego-',...
    'HorizontalAlignment','center',...
    'Fontsize',10,...
    'FontWeight','bold',...
    'Position',[340 225 100 20]);
uicontrol('Style','text',...
    'String','Imagem',...
    'HorizontalAlignment','center',...
    'Fontsize',10,...
    'FontWeight','bold',...
    'Position',[340 205 100 20]);
uicontrol('Style','text',...
    'String','Média',...
    'HorizontalAlignment','left',...
    'Fontsize',10,...
    'FontWeight','bold',...
    'Position',[80 170 200 20]);
uicontrol('Style','text',...
    'String',md_img,...
    'HorizontalAlignment','left',...
    'Fontsize',10,...
    'Position',[278 170 70 20]);
uicontrol('Style','text',...
    'String',md_img_e,...
    'HorizontalAlignment','left',...
    'Fontsize',10,...
    'Position',[365 170 70 20]);
uicontrol('Style','text',...
    'String','Desvio Padrão',...
    'HorizontalAlignment','left',...
    'Fontsize',10,...
    'FontWeight','bold',...
    'Position',[80 140 200 20]);
uicontrol('Style','text',...
    'String',dv_ori',...
    'HorizontalAlignment','left',...
```

```
        'Fontsize',10,...
        'Position',[278 140 70 20]);
    uicontrol('Style','text',...
        'String',dv_est,...
        'HorizontalAlignment','left',...
        'Fontsize',10,...
        'Position',[365 140 70 20]);
    uicontrol('Style','text',...
        'String','Coeficiente de Correlação',...
        'HorizontalAlignment','left',...
        'Fontsize',10,...
        'FontWeight','bold',...
        'Position',[80 110 220 20]);
    uicontrol('Style','text',...
        'String',crl_img_img_e,...
        'HorizontalAlignment','left',...
        'Fontsize',10,...
        'Position',[320 110 100 20]);
    verificar_callback;
```

sair_callback.m

```
function sair_callback
    close;
```

APÊNDICE C – Funções do Processo de Extração

Extrair_callback.m

```
function extrair_callback
    %
    % Cria os botoes do processo de extraçao do texto
    %
    uicontrol('Style','frame',...
        'BackgroundColor',[0.93 0.91 0.83],...
        'ForegroundColor',[0.93 0.91 0.83],...
        'Position',[500 0 205 353]);
    uicontrol('Style','pushbutton',...
        'String','Imagem',...
        'Enable','on',...
        'Fontsize',10,...
        'Position',[540 308 120 30],...
        'callback','hospedeiro_callback');
    uicontrol('Style','pushbutton',...
        'String','Extrair',...
        'Enable','on',...
        'Fontsize',10,...
        'Position',[540 258 120 30],...
        'callback','senha_callback');
    uicontrol('Style','pushbutton',...
        'String','Sair',...
        'Enable','on',...
        'Fontsize',10,...
        'Position',[540 208 120 30],...
        'callback','sair_callback');
```

hospedeiro_callback.m

```
function hospedeiro_callback
    global img_h
    global dsv
    %
    % Busca Imagem
    %
    [a_img, f_dir]=uigetfile('*.jpg', 'Escolha a foto');
    %
    % Le Imagem
    %
    img=imread([f_dir, a_img]);
    %
    % Converte a imagem em uma matriz numerica
    %
    img_h=double(img);
    %
    % Apresenta imagem escolhida na tela
    %
    subplot(1,3,1), imshow(img), title('Estego-Imagem');
    %
    % Atribui valor 2 para variavel dsv (desvio)
    % para que no processo de captaçao da senha seja
    % indicado que esta no processo de extraçao
    %
    dsv=2;
```

extrair.m

```
function extrair
    global img_h
```

```
global hash
%
% Transforma a coluna da matriz hash em linha
%
hash2=hash';
%
% Pega o tamanho da imagem
%
[M,N,O]=size(img_h);
%
% Extrai o tamanho do texto que esta no inicio do texto
%
for v=1:32,
    if mod(img_h(1,v,1),2)==0,
        aa(v)='0';
    else
        aa(v)='1';
    end
end
[x,y]=size(aa);
c=y/8;
d=1;
for i=1:c,
    for j=1:8,
        if d<(y+1),
        aaa(i,j)=aa(1,d);
        d=d+1;
        end
    end
end
h1=bin2dec(aaa(1,1:8));
j1=char(h1);
h2=bin2dec(aaa(2,1:8));
j2=char(h2);
h3=bin2dec(aaa(3,1:8));
j3=char(h3);
h4=bin2dec(aaa(4,1:8));
j4=char(h4);
n1=strcat(j1,j2);
n2=strcat(n1,j3);
nn=strcat(n2,j4);
t=str2num(nn);
%
% Extrai o texto da imagem
%
f=1;
g=1;
k=1;
for u=1:M,
    for v=1:N,
        if k==t+1,
            break;
        elseif f > N,
            f=1;
            g=g+1;
        elseif mod(img_h(g,f,1),2)==0,
            te(k)='0';
            f=f+1;
            k=k+1;
        elseif mod(img_h(g,f,1),2)~=0,
            te(k)='1';
            f=f+1;
            k=k+1;
        end
    end
end
[x,y]=size(te);
yy=y/8;
```

```
c=1;
for i=1:yy,
    for j=1:8,
        ab(i,j)=te(c);
        c=c+1;
    end
end
ad=bin2dec(ab);
au=uint8(ad);
au2=au';
%
% Decriptografar o texto
%
ext=au2;
[i,j]=size(au2);
w=(j-5)/16;
r=5;
for x=1:w,
    for y=1:16
        ext(1,r+y)=bitxor(hash2(1,y),au2(1,r+y));
    end
    r=r+16;
end
ext2=char(ext);
[l,p]=size(ext2);
ext3=ext2(6:p);
%
% Apresenta texto extraido na tela
%
uicontrol('Style','edit',...
    'String',ext3,...
    'Enable','on',...
    'Fontsize',12,...
    'Min',1,...
    'Max',18,...
    'BackgroundColor','white',...
    'HorizontalAlignment','left',...
    'Position',[15 15 485 323]);
```